VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

SCADA modul pro Cutter Information System

SCADA module for Cutter Information System

2019 Bc. David Drápela

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. David Drápela**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: SCADA modul pro Cutter Information System
SCADA Module for Cutter Information System

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je vytvoření SCADA modulu pro Cutter Information System (CIS) sloužícího jako dohledový systém umožnující sledování a práci s naměřenými fyzikálními veličinami.

1. Analyzujte současný SCADA modul, zhodnoťte jeho nedostatky.
2. Vytvořte serverovou a klientskou část modulu s funkcemi:
a) uživatel bude moci definovat veličiny, nová zařízení a jejich vstupy a výstupy,
b) zařízení bude možné organizovat do skupin,
c) serverová část modulu bude přijímat data ze zařízení ve formě SCADA dat, která jsou schopna zasílat zařízení vyrobená firmou Cutter Systems,
d) data budou zpracovávána (přepočet, vyhodnocení a dle nastavení i vyhlášení alarmu po překročení stanovených mezí),
e) data budou ukládána do souborového systému,
f) pro výstupy zařízení bude uživatel moci definovat pravidla (pokud překročí hodnota na vstupu nebo na více vstupech danou mez, bude na výstupu nastavena definovaná hodnota),
g) uživatel bude moci zobrazit pohled s přehledem všech zařízení, jejich vstupů a výstupů,
h) uživatel bude moci vytvořit vlastní pohledy (skupiny vstupů, skupiny výstupů, grafy s aktuálními daty, grafický plánek),
i) uživatel bude moci zobrazovat grafy s historickými daty, grafy bude možné tisknout,
j) uživatel bude moci pro zvolený vstup zobrazit a tisknout protokol se souhrnem dat za vybrané období,
k) data bude možné v CSV formátu exportovat.

Seznam doporučené odborné literatury:

Podle pokynů vedoucího diplomové práce.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Martin Kleiner**

Konzultant diplomové práce: Ing. David Ježek, Ph.D.

Datum zadání: 01.09.2017

Datum odevzdání: 30.04.2019

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 1. dubna 2019 .....

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.

V Ostravě 1. dubna 2019 .....

Rád bych na tomto místě poděkoval Ing. Martinu Řezáčovi a Ing. Martinu Kleinerovi ze společnosti CUTTER Systems spol. s r.o. za poskytnutí prostředků a pomoci při vývoji této práce. Dále pak rodině a přátelům za podporu během celého studia.

Abstrakt

Cílem práce je implementace SCADA modulu pro Cutter Information System, jehož nasazení se předpokládá v kombinaci s dalšími moduly v důlním průmyslu. První části práce se zabývají analýzou a návrhem řešení, využitou platformou a porovnáním se stávajícím systémem. Následuje podrobný popis technických detailů implementace a použitých technologií. Dále práce detailně přibližuje všechny vlastnosti systému z pohledu běžného uživatele. Protože je již systém zprovozněn u několika zákazníků, je závěrečná kapitola věnována následným úpravám a přizpůsobením, které vznikly na základě zkušeností provozu systému v reálném prostředí.

Klíčová slova: SCADA, monitorování veličin, modul, dispečerské pracoviště, důlní průmysl

Abstract

The objective of the thesis is to develop a SCADA module for Cutter Information System, which, among other modules, is intended to be used in the mining industry. The first part of the thesis focuses on analysis and proposed design, used platform and comparison with the existing system. Next, the work describes technical details of the implementation and used technologies. Furthermore, detailed description of the implementation including all features from the view of a regular user, is discussed. As the system has been deployed in real environment, the final part summarizes further modifications and improvements based on customers' experience.

Key Words: SCADA, sensor monitoring, module, supervisory control, mining industry

Obsah

Seznam použitých zkratek a symbolů

SCADA	–	Supervisory Control And Data Acquisition
CIS	–	Cutter Information System
IoT	–	Internet věcí
MVVM	–	Model-View-ViewModel
OPC	–	Open Platform Communication
EF	–	Entity Framework
LINQ	–	Language-Integrated Query

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Trendem poslední doby je vzrůstající zájem o zvýšení bezpečnosti na pracovištích za pomoci informačních technologií. Tomuto trendu částečně pomáhají dohledové systémy, které jsou schopny vyhodnotit riziko a s předstihem na něj upozornit. Příkladem takového systému je SCADA (dispečerské řízení sběru dat), tedy software, který komunikuje s průmyslovými zařízeními, ze kterých sbírá a vyhodnocuje data v reálném čase.

Cílem této práce je vývoj SCADA systému schopného komunikovat se zařízeními vyvinutými firmou CUTTER Systems spol. s r.o. Z těchto zařízení bude systém získávat aktuálně naměřené veličiny, které bude zpracovávat a na základě definovaných mezí bude schopen upozornit na případné riziko. Zároveň bude systém moci na základě definovaných pravidel nastavovat hodnoty na výstupech těchto zařízení.

Primární použití systému se předpokládá v důlním průmyslu nejen na území České republiky, ale i v Turecku, kde bude nasazen v kombinaci se systémem lokalizace osob. Zájem o dohledové systémy v Turecku výrazně vzrostl po sérii incidentů, z nichž nejznámější je Soma Incident[1] z roku 2014.

Úvodní kapitoly se zabývají analýzou a návrhem řešení, využitou platformou a porovnáním se stávajícím systémem. Následuje přiblížení technických detailů implementace a použitých technologií. Dále práce podrobně popisuje implementovaný systém s přiblížením všech jeho možností s přihlédnutím na budoucí rozšiřování. Protože je již systém zprovozněn u několika zákazníků, jsou poslední kapitoly věnovány následným úpravám a přizpůsobením, které vznikly na základě zkušeností provozu systému v reálném prostředí.

[1] `https://en.wikipedia.org/wiki/Soma_mine_disaster`

2 Analýza řešení

Tato kapitola stručně přibližuje obecnou charakteristiku SCADA systémů, dále popisuje motivaci pro vývoj této práce, její vlastnosti a použitou platformu.

2.1 SCADA systémy

Obecně se dá na SCADA systémy dívat jako na technologii, která umožňuje uživatelům získat data z jednoho nebo více vzdálených zařízení a zároveň na tato zařízení zasílat instrukce. SCADA systémy jsou využívány pro monitorování a ovládání zařízení, které od sebe mohou být vzdálené několik metrů, ale i desítky kilometrů a které vyžadují častou kontrolu a v případě potřeby okamžitý zásah. Příklady použití mohou být následující:

- **Ropná pole** s mnoha ropnými věžemi pokrývajících velké oblasti, u kterých je nutné pravidelně získávat informace o stavu, kdy každá věž musí být schopna rychle reagovat na podmínky ve zbytku pole.
- **Plynovody, ropovody, vodovody** mají prvky umístěné v různých vzdálenostech od řídícího střediska, které mohou být ovládány (otevírání/zavírání ventilů, ovládání kompresorů a pump).
- **Elektrické vedení** pokrývající stovky tisíc kilometrů, které může být ovládáno přepínači a kde je nutná rychlá reakce například na změny zatížení.
- **Zavlažovací systémy**, které mohou pokrývat stovky čtverečních kilometrů mohou být ovládány za pomoci ventilů.

Data ze zařízení typicky obsahují stav, alarmy a naměřené hodnoty. Instrukce zasílané ze SCADA systému jsou obvykle omezeny na binární nebo analogové hodnoty. Binární hodnoty mohou znamenat změnu stavu z „vypnuto“ na „zapnuto“, analogová hodnota může například znamenat stav otevření ventilu.[1]

Hardware SCADA systému se skládá ze vzdálených terminálových jednotek, které sbírají a přeposílají data hlavní stanici. Hlavní stanice zpracovává získaná data a poskytuje uživateli rozhraní s informacemi a ovládacími prvky pro správu vzdálených jednotek. Volitelně mohou být mezi vzdálenými jednotkami a hlavní stanici ještě další podružné stanice, které zpracovávají data z části systému a dále je předávají hlavní stanici. Komunikace může probíhat po drátu, optice, rádiu nebo satelitu.[2] Trendem poslední doby je adaptace IoT technologií, kdy zařízení může s hlavní stanicí komunikovat prostřednictvím IoT sítí (v České republice jde například o sítě *Sigfox*, *LoRa* a *NB-IoT*). S využitím těchto sítí jsou tak výrazně sníženy náklady na infrastrukturu, údržbu a integraci.

2.2 Motivace pro vývoj systému

Po získání povědomí o společnosti CUTTER Systems spol. s r.o., u níž jsem zaměstnán, v tureckém důlním průmyslu, kam se podařilo rozšířit systém pro monitorování horníků přibližně do 40 dolů, se na základě požadavků tamních zákazníků snažila firma rozšířit své portfolio. Jedním z požadavků bylo vyvinout systém pro monitorování fyzikálních veličin, konkrétně nebezpečných plynů. V případě zvýšené koncentrace těchto plynů měl systém upozornit pracovníka na dispečerském stanovišti, který by podnikl patřičné kroky pro odvrácení nebezpečí. Takový systém byl sice v určité podobě dodán jako součást původního řešení, nicméně rostoucím požadavkům zákazníků již ve své podobě nepostačoval a jeho rozšiřování bylo poměrně komplikované. Aby byly požadavky zákazníků splněny, došlo k rozhodnutí vyvinout nový systém, který by původní zcela nahradil.

Kromě využití v důlním průmyslu měl SCADA systém nahradit monitorovací systém dodaný do některých českých nemocnic, kde je využíván například pro monitorování teplot v lednicích. Aktuálně používaný systém je již zastaralý a obtížně udržovatelný, sjednocení pod jeden systém by přineslo efektivnější údržbu v případě nutnosti opravy chyb, údržby, apod.

2.3 Vlastnosti navrhovaného řešení

Základní vlastností systému je komunikace s firemními zařízeními a jejich vstupy a výstupy. Vstupem zařízení jsou data ze senzorů, která mohou být k těmto zařízením připojena, nebo jsou jejich součástí. Na výstup zařízení může systém nastavit hodnotu na základě pravidel definovaných v systému.

Jedním z takových zařízení je důlní koncentrátor *DKD11*, případně jeho odlehčená varianta *DKDIO*, k jehož vstupům je možné připojovat kombinace senzorů. Typicky je toto zařízení umístěno na několika místech v dole, kde je k němu připojena kombinace senzorů monitorujících CH_4, CO, H_2S a O_2. Na výstup tohoto zařízení může být připojen alarm, který může být spouštěn při překročení limitních mezí senzorů. Některá zařízení musí být schopna nastavovat hodnoty na výstupu i v případě výpadku serveru, a to za pomoci pravidel nahraných do zařízení z dispečerského pracoviště. Naměřená data jsou ukládána do struktury zvané *Measuring Packet*, která je blíže popsána kapitole 4.4.

Data musí být přehledně zobrazena a v případě překročení limitních mezí musí být uživatel informován jak v aplikaci, tak případně zasláním SMS nebo emailu.

Systém musí být snadno a rychle konfigurovatelný, aby s co nejmenším úsilím dokázal zobrazit všechny potřebné informace v základním přehledu. Zároveň ale musí umožnit vytvářet uživatelské pohledy, které zahrnují grafické schéma.

Ze znalosti produkčního prostředí totiž vyplynulo, že někteří uživatelé systém zprovozní pouze do takové míry, aby splnily interní směrnice, případně zákony v dané zemi, někteří uživatelé ale naopak nastavení systému věnují více času a přizpůsobí si ho svým potřebám.

2.4 Platforma

Vzhledem k faktu, že má SCADA systém fungovat v kombinaci se systémem sledování horníků, který byl vyvinut jako modul firemního frameworku CIS, bylo rozhodnuto, že SCADA systém bude postaven na stejné architektuře.

CIS je modulární systém vytvořený pro platformu Windows, jeho jádro poskytuje modulům přístup k základním funkcím, které by jinak bylo nutné implementovat v každém modulu samostatně. Mezi nejdůležitější z nich patří přístup a komunikace s databází, práce se souborovým systémem, komunikace se zařízeními, správa uživatelů a jejich práv. Podrobnějšímu popisu CIS frameworku se věnuje kapitola 3.

3 Cutter Information System

Tato kapitola popisuje Cutter Information System (dále jen CIS) do takové úrovně, aby bylo možné pochopit implementační detaily modulu SCADA. Přestože jsem se na vývoji CIS podílel, součástí práce byl pouze vývoj samotného modulu. CIS je modulární systém založený na modelu server-klient. Je napsán v jazyce C# verze 6 v běhovém prostředí .Net Framework, aktuálně ve verzi 4.5.2. Serverová část může být spouštěna jako konzolová aplikace nebo Windows služba, klientská část je desktopová aplikace pro Windows. Původně byla postavena na knihovnách Windows Forms[2], aktuální verze je postavena na WPF frameworku[3].

Primárním účelem systému je umožnit modulům komunikaci se zařízeními vyvinutými firmou CUTTER Systems spol. s r.o., jejich správu, zpracování a prezentaci dat získaných z těchto zařízení. Funkce poskytované jádrem jsou definovány rozhraním, které existuje v samostatné knihovně. Jádro jeho funkce implementuje podle vzoru separated interface[3].

Aktuálně využívaná databáze je PostgreSQL[4], nicméně systém je navržen tak, aby byl od konkrétního databázového systému odstíněn pomocí rozhraní. Implementace komunikace s databází PostgreSQL využívá pro přístup databázový ovladač Npgsql[5]. Výměna databázového systému je na úrovni jádra možná, na úrovni modulů by ale mohla být problematická, protože některé z nich využívají funkce specifické pro databázi PostgreSQL.

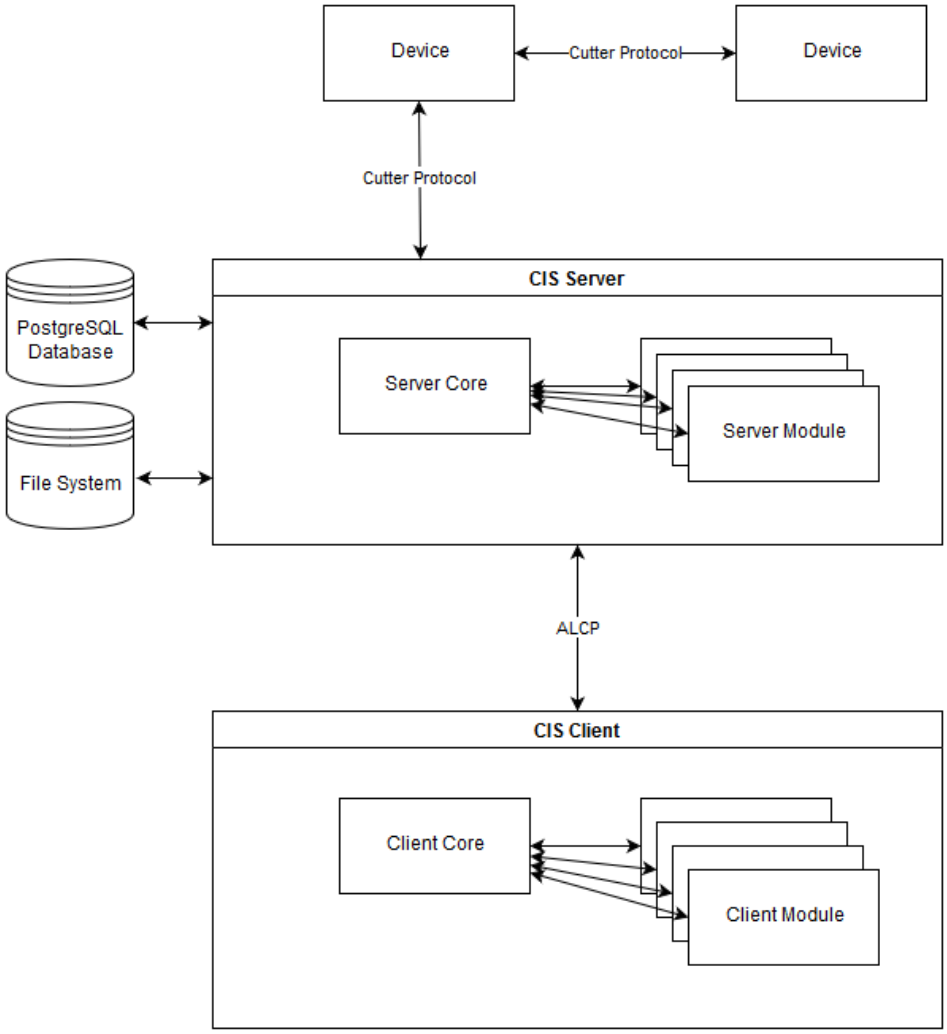
Samostatně nebo v kombinaci s jinými moduly tvoří produkty firmy CUTTER Systems, například přístupový systém, lokalizační systém pro důlní průmysl, hlasovací systém pro poslaneckou sněmovnu a další.

Klientská část aplikace sestává z hlavního okna se základním přehledem o systému a společným nastavením. Zároveň definuje několik položek menu, které zpřístupňují základní funkce, například:

- správu serveru,
- správu uživatelů a jejich práv,
- editaci profilu,
- log serveru,
- přizpůsobení okna.

Jednotlivé moduly mohou toto okno doplňovat o záložky a další položky menu podle potřeby. Obrázek č. 1 zobrazuje diagram architektury systému.

[2] `https://docs.microsoft.com/en-us/dotnet/framework/winforms/windows-forms-overview`
[3] `https://msdn.microsoft.com/en-US/library/mt149842.aspx`
[4] `https://www.postgresql.org/`
[5] Open source knihovna pro přístup k PostgreSQL databázi https://www.npgsql.org/



Obrázek 1: Architektura CIS

3.1 Komunikace se zařízeními

Komunikace probíhá pomocí protokolu **CutterProtocol** vyvinutého speciálně pro tento účel. Každé zařízení má přidělenou unikátní adresu (sériové číslo). Data jsou balena do tzv. **Cutter paketů** (dále jen paketů), jejichž hlavička nese mimo jiné informaci o adresátovi, příjemci a identifikátoru požadavku. Takto mohou být pakety směrovány přes vícero zařízení, než dorazí k cílovému zařízení. Zároveň existuje několik speciálních adres, mezi které patří:

- **Broadcast** - paket je rozeslán na všechna dostupná zařízení, toto je využíváno například pro synchronizaci času,
- **PC** - značí adresu serveru.

Základem protokolu je rozhraní komunikačního kanálu, který je implementován pro několik typů spojení. Mezi nejpoužívanější patří TCP a UPD spojení a komunikace přes sériový port.

První verze protokolu měla pro uložení adresy vyhrazenu velikost 2B. Vzhledem k rozšiřování výroby zařízení tato velikost přestala postačovat a rozsah adres byl v období vývoje této práce téměř vyčerpán. Došlo tedy k rozhodnutí navrhnout a vyvinout novou verzi protokolu, která definuje novou verzi paketu s adresou o velikosti 4B. Tato verze protokolu nicméně zůstává zpětně kompatibilní s původní verzí paketu, aby nadále fungovala komunikace s původními zařízeními.

3.2 Komunikace serveru s klienty

Server s klienty komunikuje pomocí ALCP[6] protokolu, který byl navržen primárně pro potřeby komunikace v CIS. Každý požadavek je označen identifikátorem, obě strany si mohou zaregistrovat metodu zpracovávající daný požadavek. Server tak může mít například registrovanou metodu zpracovávající požadavek přihlášení uživatele, klient naopak může zpracovávat požadavek serveru na zaslání aktuálně zvoleného jazyka.

Požadavky jsou buď globální, je možné je volat z každého modulu, nebo jsou omezeny pouze na jeden modul a pro ostatní moduly jsou nedostupné. Je tak zajištěna určitá izolace modulů od ostatních.

3.3 Popis modulu

Modul je samostatná část systému nezávislá na ostatních modulech, která se zabývá konkrétním úkolem. Modul pro CIS je implementován podle vzoru plugin[3]. Je rozdělen na serverovou a klientskou část s tím, že není nutná implementace obou částí.
Modul je typicky složen ze tří .dll knihoven:

- **serverová** - musí obsahovat třídu implementující rozhraní *IServerModule*,
- **klientská** - musí obsahovat třídu implementující rozhraní *IClientModule*,
- **sdílená** - obsahuje sdílené třídy, konstanty.

Knihovny modulů je nutné umístit do adresáře serveru nebo klienta. Jádro systému během inicializace tyto knihovny postupně prochází a vyhledává třídy implementující příslušné rozhraní. Následně vytvoří instance těchto tříd, zařadí je do seznamu modulů a volá jejich inicializační metody.

3.4 Nová generace systému

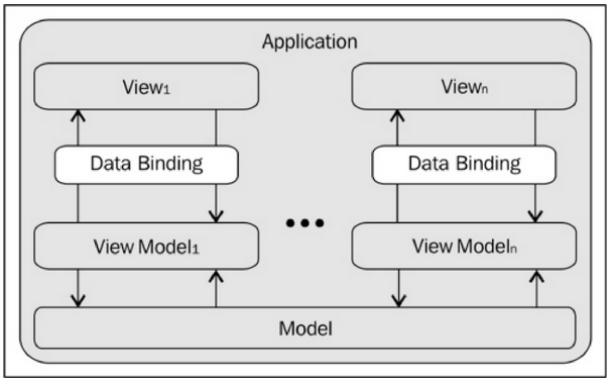
Jak již bylo zmíněno, původně byla klientská část psána v technologii Windows Forms. Protože ale bylo s použitím této technologie obtížné plnit některé požadavky zákazníků, například na grafický design, který musel být přizpůsoben konkrétnímu zákazníkovi, bylo v roce 2017 zvažováno využití jiné technologie.

[6] Abstract Layer Communication Protocol

Mezi uvažovanými možnostmi byl přechod z desktopové na webovou aplikaci, nebo volba jiné technologie pro desktopové aplikace, konkrétně WPF. Protože by první možnost vyžadovala poměrně velký zásah do celého systému a došlo by ke ztrátě zpětné kompatibility, bylo rozhodnuto o druhé variantě. Ta totiž umožnila postupný přechod se zachováním zpětné kompatibility s moduly, do kterých již není výhodné investovat větší množství času na kompletní přepsání, zároveň je ale nutné tyto moduly udržovat a opravovat chyby. WPF framework totiž obsahuje komponentu, do které je možné vkládat Windows Forms komponenty[7]. Zároveň umožňuje vývojářům navrhnout vlastní styly komponent, které je možné měnit například i za běhu aplikace.

Spolu s přechodem na novou technologii bylo také spojeno nalezení vhodného návrhového vzoru, neboť stávající vývoj probíhal podle tzv. *monolitického návrhového vzoru*[5], kde byla logika prezentační vrstvy silně provázaná s doménovou logikou, v některých případech i s přístupem k datovým zdrojům. Přestože ve WPF frameworku lze také psát podle výše uvedeného vzoru, primárně byl navržen pro vývoj podle vzoru *Model-View-ViewModel* (MVVM).

MVVM vznikl jako evoluce vzorů *MVC* a *MVP* s cílem odstranit jejich nedostatky a zároveň zkombinovat jejich silné stránky. *Model* má stejnou roli jako ve vzoru *MVP*, je tedy reprezentací dat získaných z persistentního datového úložiště. Zároveň je zodpovědný za notifikaci *ViewModelu* o změnách. *View* je zodpovědný za zobrazení dat, získávání uživatelského vstupu a jeho předávání *ViewModelu*. *ViewModel* je zodpovědný za stav a logiku *View*. Je možné jej označit jako prostředníka v *MVVM*, který se stará o přesun dat z modelu do *view* a o předávání uživatelských akcí z *view* do modelu. Provázání *View* a *ViewModelu* je zajištěno technikou *DataBinding*, která se stará o obousměrné provázání a synchronizaci dat mezi jejich poskytovatelem a odpovídajícím konzumentem. Každá změna v datech je tak ve *View* automaticky promítnuta.[5] Ukázka *MVVM* diagramu je dostupná v obrázku č. 2. Tento diagram byl převzat z [5].



Obrázek 2: Ukázka struktury MVVM

Jelikož jsem krátce po mém nástupu do firmy dostal na starost vývoj a údržbu CIS frameworku, nové jádro klientské části systému jsem navrhl a také implementoval. V této době jsem jako jediný volný vývojář v nově vznikajícím oddělení byl jediný, kdo se mohl projektem

[7]Komponenta *WindowsFormsHost*

zabývat. Prvním modulem, který měl toto nové jádro využít, byl právě SCADA systém. Vývoj jádra probíhal současně s vývojem této práce, mnoho funkcí poskytovaných jádrem bylo vyvinuto na základě potřeb práce. Klientské části nových modulů jsou již vyvíjeny výhradně s využitím nového jádra v technologii WPF.

3.5 Tabulková komponenta

Jednou ze základních funkcí systému je prezentace dat. K tomu slouží obecně použitelná tabulková komponenta *CutterGrid* vytvořená v technologii WPF pro novou generaci systému. Komponenta implementuje základní operace jako je filtrování, řazení, stránkování a editaci položek. Mezi pokročilejší funkce patří export dat do formátu CSV nebo XLSX, tisk a zobrazení dat v podobě grafů.

Umožňuje zobrazení jak lokálních dat (uložených v paměti na straně klientské aplikace), tak i dat z databázových tabulek. Pro nastavení tabulky pro zobrazení dat ve výchozím nastavení je nutné provést následující kroky:

- označit danou entitní třídu atributem *CutterGridTableAttribute*, u kterého je možné specifikovat, zda tabulka umožní zobrazení grafu
- označit požadované vlastnosti entitní třídy atributem *CutterGridColumnAttribute*, mezi nejdůležitější parametry patří:
  - název,
  - kategorie (pro seskupování, vhodné o tabulek s větším počtem sloupců),
  - výchozí pořadí sloupce,
  - nastavení, zda může být sloupec vykreslován v grafu (pokud je vykreslování povoleno na úrovni celé třídy).
- implementovat správce tabulky pro danou entitní třídu, existují dvě předchystané implementace, které je možné podědit:
  - *CutterGridManagerLocal* - pro zobrazení dat uložených v paměti,
  - *CutterGridManagerEF* - pro zobrazení dat z databáze s použitím knihovny EntityFramework.

  Pro základní použití stačí u potomků těchto tříd implementovat pouze několik vlastností vracejících texty zobrazené uživatelům, ostatní logika je již implementována na úrovni předků. Pro větší přizpůsobení chování je možné přepisovat metody předků.

4 Implementace modulu

V této kapitole je blíže popsána struktura modulu SCADA. První část se věnuje použitým technologiím, dále je čtenář seznámen s podstatnými částmi implementace práce, které zahrnují návrh datového modelu, datovou výměnu a její zpracování a návrh GUI komponent v klientské části modulu.

Pro vývoj práce bylo použito vývojové prostředí Visual Studio Professional 2015 s rozšířením ReSharper of firmy JetBrains. Pro pohodlnější psaní lokalizací bylo použito rozšíření ResXManager, které kromě doplňování textů přímo v prostředí Visual Studia umožňuje export do XLSX souboru, překlady je tedy možné psát dodatečně.

4.1 Použité technologie

Tato kapitola popisuje použité technologie mimo CIS framework a další podnikové knihovny. Jedná se převážně o knihovny třetích stran, patří mezi ně například obecně použitelné GUI komponenty, grafy, ORM.

4.1.1 Entity Framework

Entity Framework[8] (dále jen EF) je ORM[9] vyvinutý společností Microsoft pro platformu .Net. ORM framework slouží k synchronizaci mezi dvěma různými reprezentacemi dat (relační DB vs objekty). Poskytuje tak vývojářům určitou míru abstrakce, protože jsou odstínění od samotného přístupu k databázi, pracují pouze s objekty definovanými v doménové vrstvě.

EF není omezen pouze na podporu databázového systému MSSQL, podporuje i systémy MySQL, PostgreSQL, SQLite, nově i in-memory databázi vhodnou pro testovací účely.

Aktuálně používaná verze v rámci CIS a tedy i SCADA modulu je 6.1, do budoucna je plánován přechod na novou verzi EF Core, která je implementována pro .Net Core aplikace. EF Core verze je de facto kompletní reimplementace frameworku a zatím neobsahuje všechny funkce z původní verze.

4.1.2 Fody

Fody[10] je open-source implementace aspektově orientovaného paradigma (AOP), jehož cílem je oddělení tzv. průřezových problémů, tedy opakující se funkcionalitou používanou napříč celým projektem. Příkladem takového průřezového problému je logování. To může být použito například v uživatelském rozhraní, business logice, přístupu k databázi a tak dále. I v rámci samostatné vrstvy může být logování použito napříč více třídami a službami[4]. Cílem je tedy upravit chování programu vložením kódu na definované místo, například na začátek a na konec metod.

[8] `https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/ef/overview`
[9] Object Relational Mapping
[10] `https://github.com/Fody/Fody`

Knihovna Fody je pouze framework, ke kterému je možné přidávat doplňky zaměřující se na konkrétní úpravy chování programu. Jedním takovým doplňkem, který je používán v modulu SCADA, je PropertyChanged.Fody[11], který automaticky do všech tříd implementujících rozhraní *INotifyPropertyChanged* doplní ve všech setterech vlastností volání události *PropertyChanged*. Tato událost je důležitá v GUI vrstvě, která je pomocí této události upozorněna, že došlo ke změně vlastnosti ve třídě (například ve ViewModelu) a že je nutné na tuto změnu reagovat, například překreslením komponenty. Ruční vkládání vyvolání události by způsobilo zbytečné nabytí velikosti tříd, protože by, mimo jiné, nebylo možné použít automaticky implementované vlastnosti[12]. Porovnat rozdíl je možné v následující ukázce (s ohledem na velikost ukázky byla vynechána samotná implementace metody *OnPropertyChanged* vyvolávající událost *PropertyChanged*).

```
//definice bez pouziti PropertyChanged.Fody
public class A : INotifyPropertyChanged
{
   private int c;
   public int C
   {
      get { return c; }
      set
      {
         c = value;
         OnPropertyChanged(nameof(C));
      }
   }
}

//definice s pouzitim PropertyChanged.Fody
public class B : INotifyPropertyChanged
{
   public int C { get; set; }
}
```

Výpis 1: Ukázka použití Aspect Weaveru PropertyChanged.Fody

[11] `https://github.com/Fody/PropertyChanged`

[12] V některých případech jsou gettery a settery vlastností použity jen pro získání nebo přiřazení hodnoty lokální proměnné. S pomocí automaticky implementovaných vlastností je tato proměnná generována kompilátorem automaticky.[7]

4.1.3 MahApps.Metro

MahApps.Metro je open-source UI toolkit, který mění vzhled stávajících komponent na styl Metro[13], zároveň však přidává několik vlastních komponent, které chybí v základním balíku komponent, například DateTimePicker nebo NumericUpDown.

4.1.4 Extended WPF Toolkit

Extended WPF Toolkit[14] je open-source knihovna obsahující kolekci UI komponent pro WPF, patří mezi ně i komponenta AvalonDock využita v modulu SCADA. AvalonDock je komponenta umožňující jednoduché přizpůsobení obsahu uvnitř pomocí dokovacího systému, který je populární zejména u vývojových prostředí jako například Visual Studio.[8]. V modulu Scada je použit pro uživatelské přizpůsobení pracovní plochy podle potřeb daného pracoviště.

4.1.5 LiveCharts

Pro vykreslování grafů je v modulu SCADA použita open-source knihovna LiveCharts[15]. Přestože v době psaní této práce ještě nebyla plně dokončena, svými možnostmi překonává všechny prozkoumávané knihovny tohoto typu pro technologii WPF. Původně byla v práci použita knihovna DynamicDataDisplay od společnosti Microsoft, ta je ale již delší dobu neudržovaná a obsahovala chyby, které v některých případech způsobovaly pád aplikace (například v kombinaci s komponentou AvalonDock).

4.1.6 Json.NET

Json.Net[16] je framework pro práci s daty ve formátu JSON. V modulu SCADA je použit zejména pro ukládání rozložení objektů v grafickém schématu. Použití tohoto frameworku nahradilo v průběhu vývoje práce ukládání rozložení ve formátu XML, zejména pro jednodušší použitelnost.

4.1.7 Dynamic LINQ

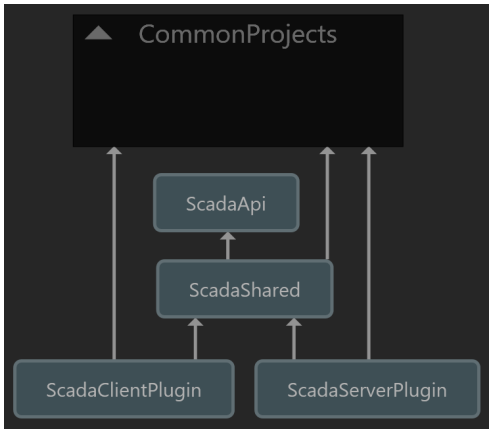
Language-Integrated Query je inovace.NET Frameworku zavedená ve verzi 3.5, která spojuje svět objektů a svět dat.[7]. LINQ je integrovaný jazyk, který sjednocuje přístup k datům z různých zdrojů - databáze, XML dokumenty, interní kolekce. Jak už název vypovídá, knihovna Dynamic LINQ umožňuje dynamické generování LINQ dotazů z textových řetězců[6]. Toto je výhodné v případech, kdy je nutné vytvářet dotaz za běhu aplikace, například podle nastavení uživatele. Nevýhodou je ztráta typové kontroly při sestavení aplikace. V CIS frameworku a částečně i ve SCADA modulu je Dynamic LINQ použit pro generování dotazů v tabulkových pohledech,

[13] `https://en.wikipedia.org/wiki/Metro_(design_language)`
[14] `https://github.com/xceedsoftware/wpftoolkit`
[15] `https://lvcharts.net/`
[16] `https://www.newtonsoft.com/json`

kde je možné z klientské části aplikace flexibilně generovat dotazování nad jakoukoliv registrovanou entitní třídou v databázi.

4.2 Struktura projektu

Práce je rozdělena do 4 částí (knihoven). Kromě samotné implementace serverové a klientské části byly vytvořeny podpůrné knihovny, které jsou využity v obou částech modulu. Struktura projektu je zobrazena diagramem v příloze č. 3.



Obrázek 3: Diagram závislostí projektu

4.2.1 ScadaApi

Scada Api obsahuje definice rozhraní popisující nejdůležitější části datového modelu. Konkrétně se jedná o definice zařízení, senzorů a výstupních pravidel. Zároveň definuje rozhraní pro přístup k těmto definicím. Tato knihovna vznikla dodatečně pro potřeby přístupu k definicím z jiných modulů.

4.2.2 ScadaShared

Podpůrná knihovna ScadaShared obsahuje definice jednotlivých tříd datového modelu spolu s definicemi rozhraní pro jejich správu. Kromě implementace datového modelu je v této knihovně implementována většina textů použitých napříč prací. Práce byla navrhována jako vícejazyčná a obsahuje české a anglické texty. Každá lokalizace je uložena v samostatném souboru, překlady do dalších jazyků je možné tedy jednoduše přidávat dle potřeby.

4.2.3 ScadaServer

Serverová část modulu implementuje výkonnou část celé aplikace. Mezi její zodpovědnosti patří zpracování přijatých dat ze senzorů, vyhodnocení výstupních pravidel, správa datového modelu a komunikace s databází, zpracování požadavků z klientské části modulu. Struktura projektu je následující:

- **Kořenový adresář** - definice hlavních tříd serverové části modulu,
- **ClientRequestHandlers** - definice tříd zpracovávající požadavky z klientské části modulu,
- **Communication** - implementace komunikaci se zařízeními,
- **DataProcessing** - implementace zpracování a vyhodnocení dat,
- **Model** - implementace rozhraní pro práci s datovým modelem ,
- **Queries** - definice SQL dotazů (pro použití mimo EntityFramework),
- **Structures** - definice pomocných struktur používaných napříč serverovou částí modulu.

Všechny části byly navrženy s ohledem na budoucí rozšiřování systému, zejména na přidávání nových zdrojů dat. Návrh také počítá s propojením s dalšími moduly v rámci CIS frameworku, ale i s propojením s externí systémy, ze kterých SCADA může data přijímat, ale může je do nich i odesílat.

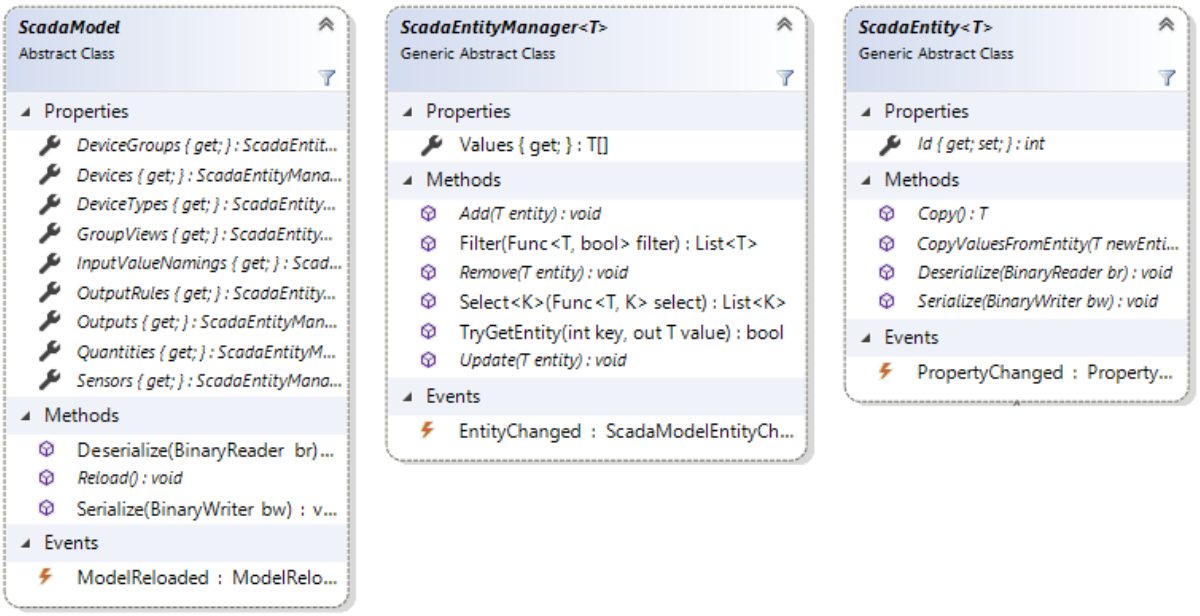
4.2.4 ScadaClient

Knihovna klientské části modulu představuje prezentační vrstvu a nevykonává žádnou logiku. Přesto ale byla časově náročnější na návrh a odladění vzhledem k množství pohledů, které uživatelům poskytuje. Struktura této části je následující:

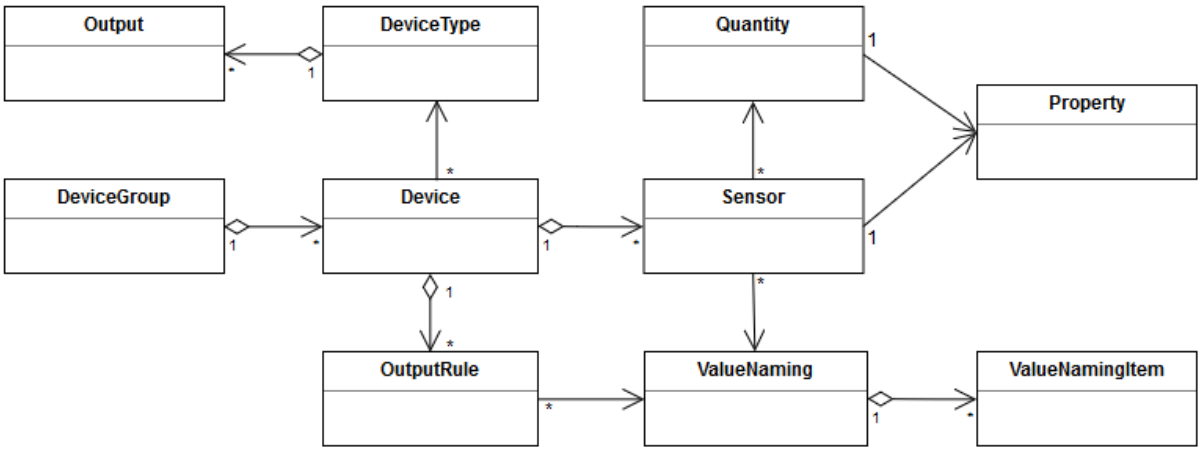
- **Kořenový adresář** - definice hlavních tříd klientské části modulu,
- **Alerts** - implementace zpracování a prezentace výstrah,
- **CommonComponents** - společné GUI komponenty využívané v různých pohledech,
- **Converters** - definice konvertorů používaných v GUI komponentách,
- **GridManagers** - implementace všech tabulkových pohledů,
- **DockWindows** - implementace všech komponent, které je možné vytvořit na pracovní ploše,
- **Graphs** - implementace grafových pohledů,
- **Model** - implementace rozhraní pro práci s datovým modelem,
- **Protocols** - implementace zobrazení protokolu,
- **SchemaDesigner** - implementace návrháře grafického schématu,
- **Styles** - definice společných GUI stylů,
- **Views & ViewModels** - definice hlavního pohledu SCADA modulu a dalších pohledů.

4.3 Datový model

Třídy datového modelu jsou definovány ve sdíleném projektu *ScadaShared*, kde zároveň existují i definice rozhraní pro jejich správu (abstraktní třída *ScadaEntityManager*). Tato rozhraní jsou definována po vzoru Table Module[3] a jsou implementována serverovou i klientskou částí modulu, je tak zajištěn jednotný přístup ze všech částí práce. Přístup k datovému modelu je možný pomocí singletonu *ScadaModel*. Všechny třídy datového modelu vycházejí z abstraktní třídy *ScadaEntity* definující společné metody využívané ve vyšších vrstvách, například jejich serializace. Pro lepší představu výše zmíněného je v příloze č. 4 k nahlédnutí diagram těchto abstraktních tříd, v příloze č. 5 je pak k nahlédnutí diagram jednotlivých tříd datového modelu.



Obrázek 4: Diagram abstraktních tříd pro práci s datovým modelem



Obrázek 5: Diagram tříd datového modelu

V serverové části modulu datový model pracuje přímo s databází za pomoci knihovny *Entity Framework*. Klientská část pracuje s daty právě prostřednictvím serverové implementace. Interně mezi oběma částmi probíhá automatická synchronizace, ostatní části práce využívající datový model mají tedy automaticky zajištěn přístup k aktuálním definicím. O jakékoliv změně jsou zároveň notifikovány v případě přihlášení k událostem datového modelu (vzor *publish-subscribe*).

4.4 Datová výměna

Komunikace mezi CIS serverem a zařízeními probíhá pomocí firemní knihovny CutterProtocol, která je popsána v kapitole 3.1. Samotná data ze vstupů a výstupů jsou zasílána ve struktuře zvané *Measuring Packet*, která je vkládána do datové části cutter paketu. Tato struktura je popsána v tabulce č. 1. Návrh datové struktury vznikl již dříve při vývoji původního systému, pro účely modulu SCADA byl rozšířen o nové typy, mezi nejvýznamnější z přidaných typů patří informace o hodnotě na výstupech zařízení.

Název	Velikost	Obsah
Délka dat	1B	délka datové části bez CRC
Typ záznamu (RT)	1B	typ záznamu podle tabulky níže
Adresa odesílatele	2B nebo 4B	adresa zařízení, velikost dle RT
Datum a čas	4B	UTC čas v sekundách od 1.1.2000
Data	(n)B	data podle specifikace RT
Kontrolní byte datové části	1B	CRC

Tabulka 1: Struktura Measuring Packet

Obsah datové části se liší podle typů záznamu, typy používané ve SCADA modulu jsou uvedeny v tabulce č. 2. Jak je z této tabulky patrné, její druhá polovina pouze rozšiřuje typy definované v první polovině. Toto rozšíření je důsledkem změny komunikačního protokolu, v rámci které došlo k rozšíření velikosti adresy zařízení (viz kapitola 3.1).

Typ záznamu (RT)	Popis
0x01	Základní typ obsahující hodnoty na vstupech
0x04	Rozšiřující typ pro modbusová čidla
0x05	Rozšiřující typ pro rozšiřující moduly k zařízení
0x07	Základní typ obsahující hodnoty na výstupech
0x81	Rozšíření typu 0x01 pro novou verzi CutterProtocol
0x84	Rozšíření typu 0x04 pro novou verzi CutterProtocol
0x85	Rozšíření typu 0x05 pro novou verzi CutterProtocol
0x87	Rozšíření typu 0x07 pro novou verzi CutterProtocol

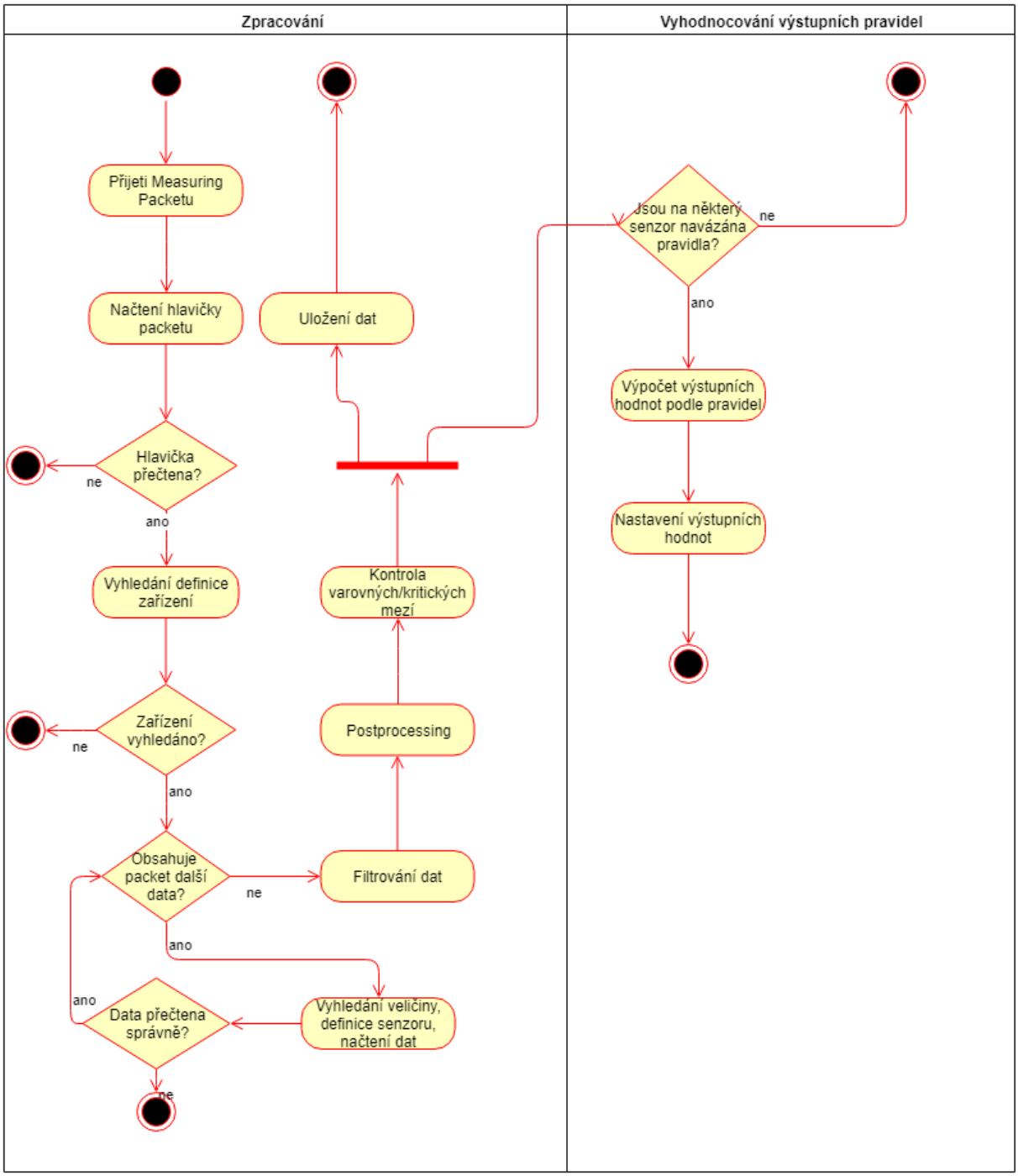
Tabulka 2: Datová část Measuring Packet

Motivací pro vznik typů záznamů byla rozdílná složitost identifikace veličiny a její aktuální hodnoty v rámci zařízení. Například typ RT1 zasílá pouze řadu dvojic ve formátu *veličina:hodnota*, což může stačit u jednoduchého zařízení, které měří teplotu a vlhkost. Naopak důlní koncentrátor DKD11 může mít připojeny až 4 modbusové linky, v rámci kterých může být sériově připojeno několik senzorů, kdy některé z nich mohou měřit stejnou veličinu a je tedy nutné ještě specifikovat port, ke kterému je modbusová linka připojena, včetně pořadí senzoru. Pro tento účel vznikl typ 0x04. Typ 0x05 slouží pro identifikaci veličiny zaslané z rozšiřujícího modulu DKD11. Typ RT7 se od ostatních liší nejvíce, protože místo měřené veličiny obsahuje jednoznačné označení výstupu pro daný typ zařízení včetně jeho typu (binární, analogový) a informace o tom, zda jeho aktuální hodnota byla nastavena vynuceně (zásahem uživatele, kterým došlo k přepsání hodnoty na základě nastaveného pravidla).

Zpracováním dat z různých zdrojů se zabývají implementace abstraktní třídy *DataProcessingManager*. Tato třída definuje události *SensorDataArrived* a *OutputDataArrived*, kterými je zbytek systému notifikován o přijetí nových dat ze zařízení. Zároveň poskytuje podpůrné metody pro vyhledávání definic pro správnou identifikaci a načtení těchto dat. Konkrétním příkladem implementace je třída pro zpracování naměřených hodnot uložených ve struktuře *Measuring Packet*, jejíž proces od přijetí paketu až po následný postprocessing je ve zjednodušené podobě znázorněn v diagramu na obrázku č. 6. Dalším příkladem implementace je simulátor pro generování dat ze senzorů, případně dodatečně implementované zpracování dat z *PLC Amit*, které je zmíněno v kapitole 7.

Na události o přijetí dat je navázáno následné zpracování, ve kterém jsou nově přijatá data filtrována a validována. Dále dochází na základě načtené veličiny k přepočtu hodnoty a kontrole, zda nedošlo k překročení varovných nebo kritických mezí. Pokud jsou na senzor vázána serverová pravidla, dojde k přepočtu a nastavení hodnoty na výstupech. V případě přijetí výstupních dat je celý proces zjednodušen, neboť tato data slouží pouze pro informativní účely, nejsou na ně tedy vázána žádná další pravidla.

Ukládání naměřených dat bylo původně zvažováno přímo v databázi PostgreSQL spolu se všemi definicemi, vzhledem k předpokládanému objemu záznamů (u standardní instalace s 50 senzory může jít o jednotky miliard ročně) bylo od této možnosti upuštěno. Aktuální implementace využívá pro ukládání dat souborový systém, kde jsou data roztříděna do složek podle identifikátoru senzoru, následně pak podle roku a měsíce. Záznamy z jednotlivých dnů jsou ukládány do binárních souborů, každý záznam se skládá ze 12B - 4B pro uložení času v sekundách, 8B pro samotnou hodnotu. Tento způsob oproti ukládání dat v relační databázi přináší několik nevýhod a omezení. Systém si pro každý senzor ukládá časovou značku aktuálních dat, pokud ale z nějakého důvodu přijdou ze zařízení data se starší časovou značkou, jsou tato data zahozena, neboť systém zapisuje pouze na konec aktuálního binárního souboru. Výhodou tohoto způsobu je snadná manipulace s daty, například zálohování, případně odstranění starých naměřených dat je otázkou přesunutí složky obsahující naměřená data za libovolný měsíc, případně rok a zvládne ji provést i zaškolený uživatel bez znalosti relačních databází.



Obrázek 6: Diagram zpracování dat

4.5 Pracovní plocha

Systém pracovní plochy je založen na komponentě *AvalonDock* z knihovny *Extended WPF Toolkit*, viz kapitola 4.1.4. Pracovní plocha se skládá z panelů s pohledy, které si může uživatel libovolně umístit, například i do samostatného okna. Každý panel je definován dvěma třídami. Základní částí je *ViewModel* vycházející ze třídy *DockWindowViewModel*, která definuje základní vlastnosti panelu včetně metod pro uložení a načtení nastavení. Nastavení je ukládáno v *JSON* formátu a spolu s nastavením rozložení komponenty *AvalonDock* je odesíláno na server. Každý panel je jednoznačně identifikován unikátním ID využívaném zejména při použití zdrojů, které nejsou ukládány jako součást nastavení. Typickým příkladem použití externích zdrojů je pozadí panelu s grafickým schématem pracoviště, kdy je pozadí samostatně načítáno při inicializaci panelu.

Ke každému *ViewModelu* musí existovat *View* definující vzhled panelu. Párování odpovídajícího *View* k *ViewModelu* je implementováno za pomoci třídy *DataTemplateSelector*, který na základě typu datového objektu umí zvolit správnou šablonu.[7]. Diagram tříd panelů je zobrazen v obrázku č. 7.

Obrázek 7: Diagram tříd panelů pracovní plochy

Jednotlivé implementace panelů z pravidla nemají přímé reference na třídy datového modelu, neboť jejich vlastnosti nemusí obsahovat informace v prezentovatelné podobě (například mohou pouze obsahovat identifikátory na položky z číselníků). Využívají tedy korespondující *ViewModel* verze těchto tříd, které interně transformují všechny potřebné informace pro prezentaci a je tak možné na ně pohodlně vytvořit přímé vazby z *View* za pomoci techniky *DataBinding*.

4.6 Schéma

Schéma (plánek) je samostatný panel umožňující graficky vizualizovat monitorované pracoviště. Schéma by se dalo označit jako jednu z nejnáročnějších komponent systému, jejíž vývoj byl oproti ostatním částem výrazně časově náročnější. Protože při vynechání této komponenty by systém stále splnil zadání a byl by plně použitelný, přišla při vývoji na řadu až jako poslední.

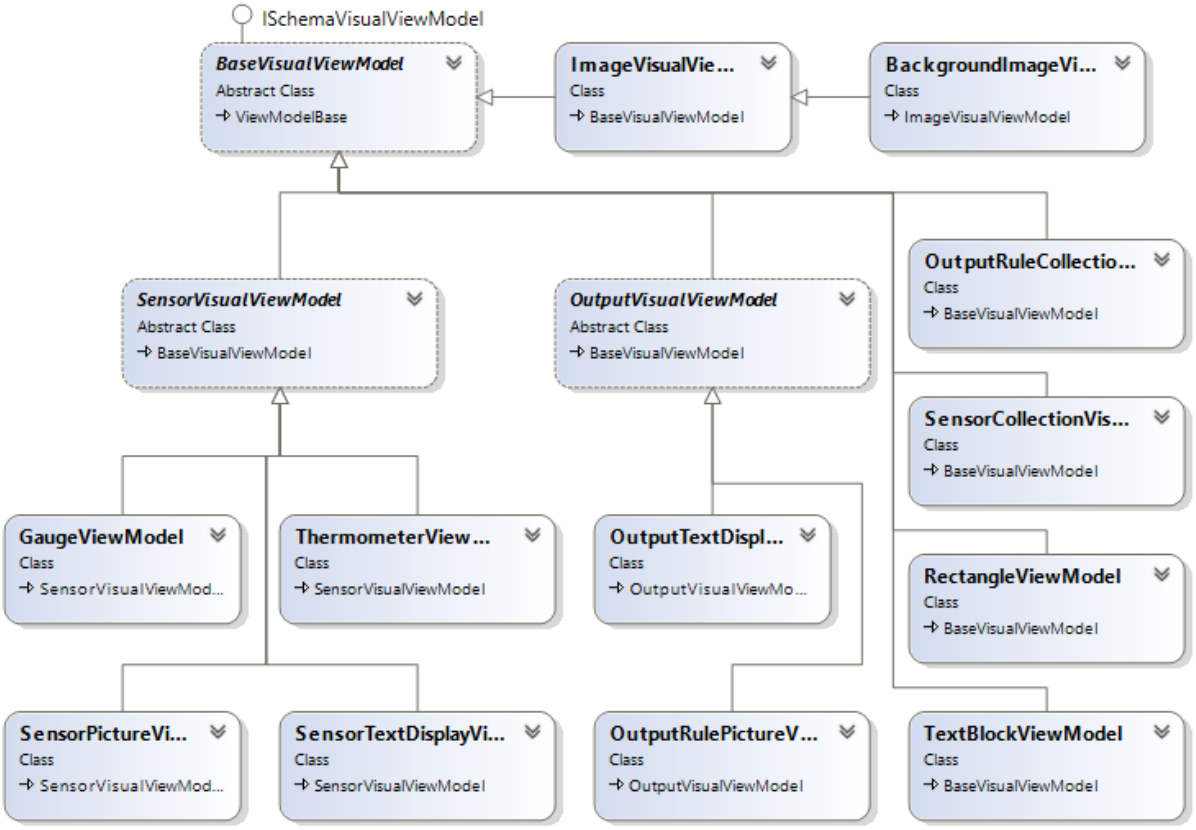
Základem schématu je třída *SchemaCanvas*, která představuje plátno, do kterého jsou vykreslovány jednotlivé grafické prvky. Stará se o správu a manipulaci těchto prvků, zároveň poskytuje podporu pro změnu velikosti (přiblížení/oddálení) a posouvání plátna. Základem všech grafických prvků je abstraktní třída *BaseVisualViewModel*, která definuje vlastnosti popisující tvar prvku:

- pozici (X,Y),
- Z-Index,
- výšku a šířku,
- úhel,
- měřítko,
- průhlednost.

Zároveň definuje metody pro transformaci, které jsou používány ze třídy *SchemaCanvas*. Jednotlivé implementace grafických prvků je možné rozdělit do 3 kategorií:

- obecné statické objekty,
- objekty vizualizující jeden nebo více senzorů,
- objekty vizualizující jeden nebo více výstupů.

Příkladem statických objektů může být pozadí schématu, ale i neměnný informativní text nebo rámeček. Ostatní kategorie mohou zahrnovat dynamicky měnící se text nebo obrázek na základě aktuální hodnoty senzoru nebo výstupu, případně i vizualizace budíku nebo teploměru. Pro lepší představu je v obrázku č. 8 přiložen diagram tříd.



Obrázek 8: Diagram grafických prvků schématu

5 Popis funkčnosti systému

V této části je popsán systém z pohledu běžného uživatele. Protože je modul SCADA postaven na frameworku CIS, je nejprve popsána práce s tímto frameworkem, následně je dopodrobna probráno ovládání samotného modulu.

5.1 Základ práce s CIS frameworkem

Tato kapitola stručně shrnuje práci s CIS frameworkem. Podrobnější manuál by značně navýšil počet stránek této práce, čtenář je tedy pouze seznámen s přehledem nejdůležitějších funkcí, kdy u každé z nich je uveden stručný popis. Po přečtení této kapitoly by měl být čtenář schopen pracovat s klientskou částí CIS do takové míry, aby mohl snadno prozkoumat a porozumět všem funkcím modulu SCADA.

5.1.1 Systémové požadavky

Systémové požadavky se mohou lišit podle velikosti instalace, systém komunikující se stovkami zařízení má vyšší nároky na hardware než prezentační instalace s několika připojenými zařízeními. Minimální systémové požadavky jsou následující:

CIS Server
Podporované operační systémy:

- Windows 7 SP1 nebo novější,
- Windows Server 2008R2 nebo novější.

Hardware:

- Dvoujádrový procesor 1,8 GHz nebo rychlejší,
- 2 GB paměti RAM,
- 5 GB místa na pevném disku, závisí na velikosti instalace.

Další požadavky:

- .NET Framework 4.5.2 nebo novější,
- databáze PostgreSQL 9.5 nebo novější,
- libovolný prohlížeč SQLite databáze (pro editaci konfiguračního souboru).

CIS Klient

Podporované operační systémy:

- Windows 7 SP1 nebo novější,
- Windows Server 2008R2 nebo novější.

Hardware:

- Dvoujádrový procesor 1,8 GHz nebo rychlejší,
- 2 GB paměti RAM,
- 1 GB místa na pevném disku.

Další požadavky:

- .NET Framework 4.5.2 nebo novější.

5.1.2 Instalace

Tato kapitola popisuje instalaci klientské části CIS frameworku. Instalace serverové části je popsána v samostatné kapitole (6), neboť je obsáhlejší a vyžaduje základní znalosti práce s databázovým systémem PostgreSQL a znalosti správy služeb v systému Windows. Instalace CIS Klienta probíhá pomocí dodaného instalačního průvodce[17]. V jednom z kroků je uživatel vyzván ke zvolení adresáře, do kterého bude aplikace instalována. Je nutné zvolit takový adresář, do kterého má aktuální uživatel Windows povoleno právo zapisovat. Aplikace v tomto adresáři vytváří soubory s logy, případně může ze serveru stahovat dočasné soubory.

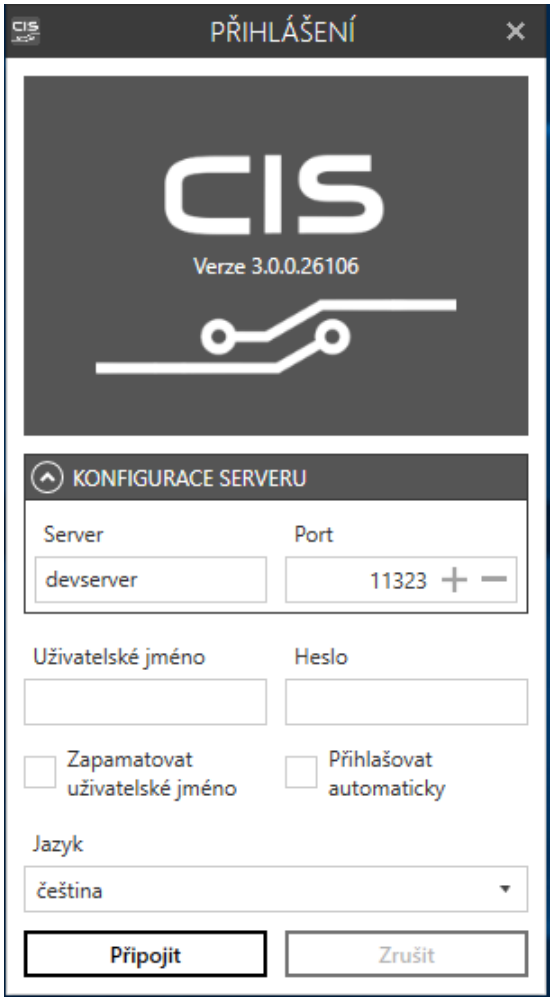
5.1.3 Přihlášení

Spuštění CIS Klienta je možné provést poklikáním na zástupce vytvořeného při instalaci, případně spuštěním souboru *ISClientWPF.exe* v adresáři instalace.

Po spuštění aplikace se zobrazí přihlašovací dialogové okno. V případě prvního přihlášení je nutné nakonfigurovat IP adresu serveru a Port. Dále je nutné zadat uživatelské jméno a heslo. V případě zaškrtnutí pole *Zapamatovat uživatelské jméno* dojde k jeho zapamatování při příštím přihlášení, v případě zaškrtnutí *Přihlašovat automaticky* dojde i k zapamatování hesla, kdy při příštím spuštění aplikace je po uplynutí desetisekundového časového intervalu uživatel automaticky přihlášen. V případě, že se jedná o první přihlášení pod uvedeným uživatelským účtem, je uživatel vyzván k zadání nového hesla.

Po připojení k serveru systém automaticky zkontroluje dostupné aktualizace a případně stáhne aktuální verzi. Po vydání nové verze systému tedy není nutné CIS Klienta přeinstalovat.

[17] Instalátor byl sestaven za pomoci programu Inno Setup



(a) Dialogové okno přihlášení

(b) Okno průběhu přihlášení

Obrázek 9: Přihlášení do CIS

5.1.4 Hlavní obrazovka

Po úspěšném přihlášení se zobrazí hlavní okno celé aplikace. V horní části je zobrazeno menu, pod ním je zobrazen systém záložek. Jádro CIS Klienta poskytuje některé položky menu pro přístup k základní správě systému, případně k funkcím používaných ve více modulech. Níže je uvedena struktura menu poskytovaného jádrem CIS Klienta, u některých položek je uveden popis:

- **Systém**
  - **Uživatelský profil** - zobrazí okno umožňující spravovat některé údaje uživatelského účtu
  - **Administrace** - obsahuje další menu umožňující spravovat nastavení systému
    - **Správa uživatelů** - zobrazí okno s hierarchií uživatelů a jejich práv

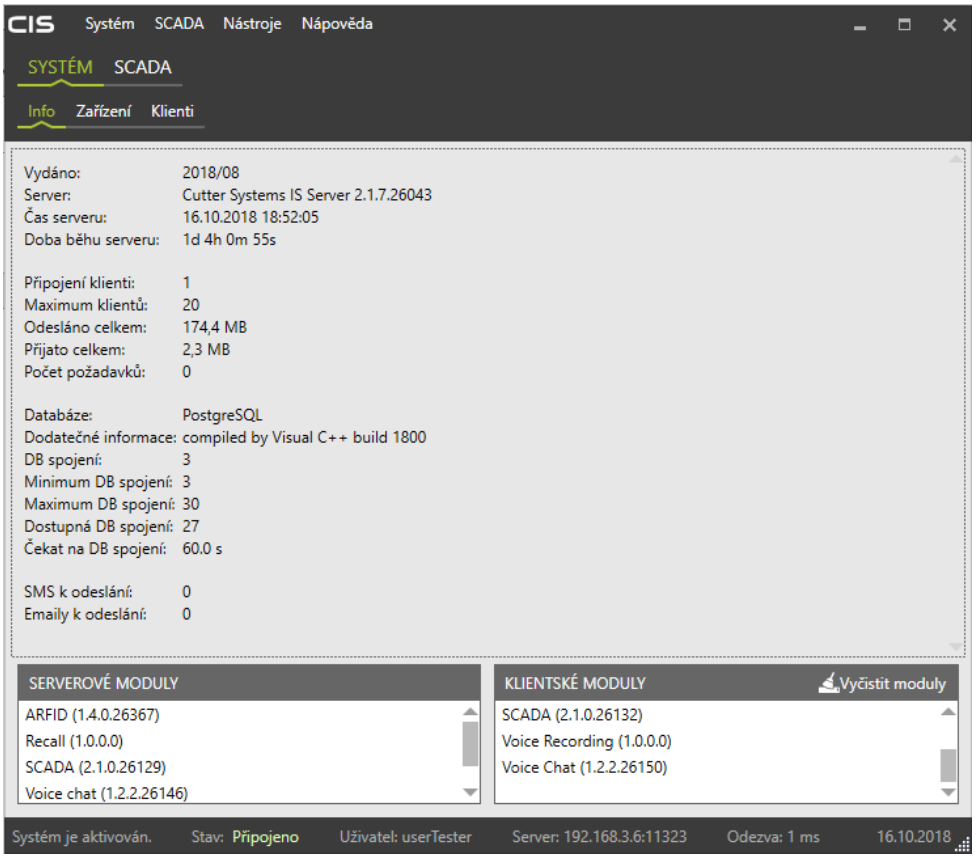
        * **Správa emailových skupin**
        * **Odeslané SMS** - pokud CIS využívá GSM modul k zasílání notifikací (například alarmů), jsou zde logovány všechny odeslané SMS
        * **Správa serveru** - zobrazí okno umožňující modifikovat nastavení serveru
        * **Log serveru** - zobrazí okno s aktuálním logem serveru
    - **Výstrahy** - obsahuje další menu umožňující spravovat a zobrazovat výstrahy (výstrahy týkající se modulu SCADA jsou popsány v kapitole 5.10)
        * **Zvuk** - zapnutí/vypnutí přehrávání varovného zvuku při aktivní výstraze
        * **Notifikační okno** - umožňuje nastavit zobrazení notifikačního okna výstrah
        * **Zobrazit výstrahy** - zobrazí okno s aktuálně vyhlášenými výstrahami
        * **Historie výstrah** - zobrazí okno s historií výstrah
        * **Akce** - zobrazí okno s nastavením reakcí na vyvolání zvolených výstrah, například při výpadku komunikace se zařízením je možné nastavit zaslání emailu správci systému
    - **Přizpůsobení**
        * **CSV Export** - zobrazí obecné nastavení exportu do CSV
        * **Ukončit aplikaci zavřením okna** - pokud je zaškrtnuto, je aplikace po zavření okna ukončena, pokud není zaškrtnuto, je pouze skryta na pozadí a je možné ji spustit pomocí ikony z notifikačního okna
        * **Zobrazovat zprávy v notifikační liště**
        * **Vyčistit moduly** - vyčistí všechny aktuálně stažené CIS moduly a restartuje CIS, může řešit vyřešit problém, kdy načtení některého z modulů opakovaně selhává
    - **Odhlásit** - odhlásí uživatele a zobrazí přihlašovací dialog
    - **Ukončit** - ukončí aplikaci
- **Nástroje** - obsahuje pomocné funkce pro správce systému
    - **Packet Forwarding** - zobrazí okno s ovládáním funkce předávání paketů, která umožňuje nastavit přeposílání paketů z libovolné aplikace konkrétnímu připojenému zařízení prostřednictvím CIS Serveru (typicky je použito například u samostatné aplikace pro přehrávání firmware v zařízeních)
- **Přehrávání firmwaru** - zobrazí okno s nastavením aktualizací firmwaru v zařízeních připojených k serveru přímo z CIS frameworku (náhrada samostatné aplikace zmíněné předchozím bodě)

- **Nápověda**
  - **O aplikaci**
  - **Licence** - zobrazí informace o aktuálně použité licenci
  - **Prohlášení o zabezpečení**

Kromě položky menu poskytuje jádro CIS Klienta první záložku s obecným přehledem systému. Tato záložka je dále rozdělena na 3 části:

- **Info** - obsahuje základní informace o stavu systému, ve spodní části jsou uvedeny aktuálně načtené moduly na straně serveru i klienta,
- **Zařízení** - obsahuje tabulku s přehledem aktuálně připojených zařízení a jejich stavu, zároveň je v této tabulce možné zobrazit konfigurační okno jednotlivých zařízení, případně zahájit aktualizaci firmwaru,
- **Klienti** - obsahuje tabulku s přehledem připojených klientů.

Zobrazení výše uvedených částí je konfigurovatelné pomocí uživatelských práv.



Obrázek 10: Hlavní okno CIS frameworku

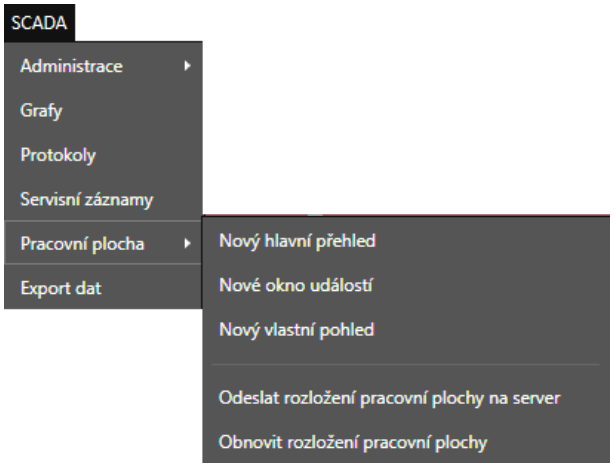
5.2 Úvod k modulu SCADA

Modul SCADA rozšiřuje CIS Klienta o záložku s pracovní plochou a položku menu umožňující přístup k administraci a dalším funkcím.

Pracovní plocha představuje virtuální dispečerské pracoviště umožňující spravovat a monitorovat stav připojených zařízení, zejména naměřené fyzikální veličiny. K tomu slouží série přehledů, které si může uživatel rozložit na pracovní ploše podle potřeby. Tyto přehledy je možné libovolně umístit, může tak například vzniknout systém záložek, kdy každá obsahuje vlastní přehled, nebo jich může být více umístěno vedle sebe. V případě potřeby je možné přehled vytáhnout z pracovní plochy a zobrazit jej jako samostatné okno.

5.3 Práce s pracovní plochou

Spravovat pracovní plochu je možné pomocí položky *Pracovní plocha* v hlavním menu SCADA modulu, viz obrázek č. 11. Rozložení pracovní plochy je sdíleno se všemi uživateli. Pro správu pracovní plochy musí mít uživatel oprávnění **Správa pracovní plochy**. Popis jednotlivých možností je uveden níže:



Obrázek 11: Menu s nastavením pracovní plochy

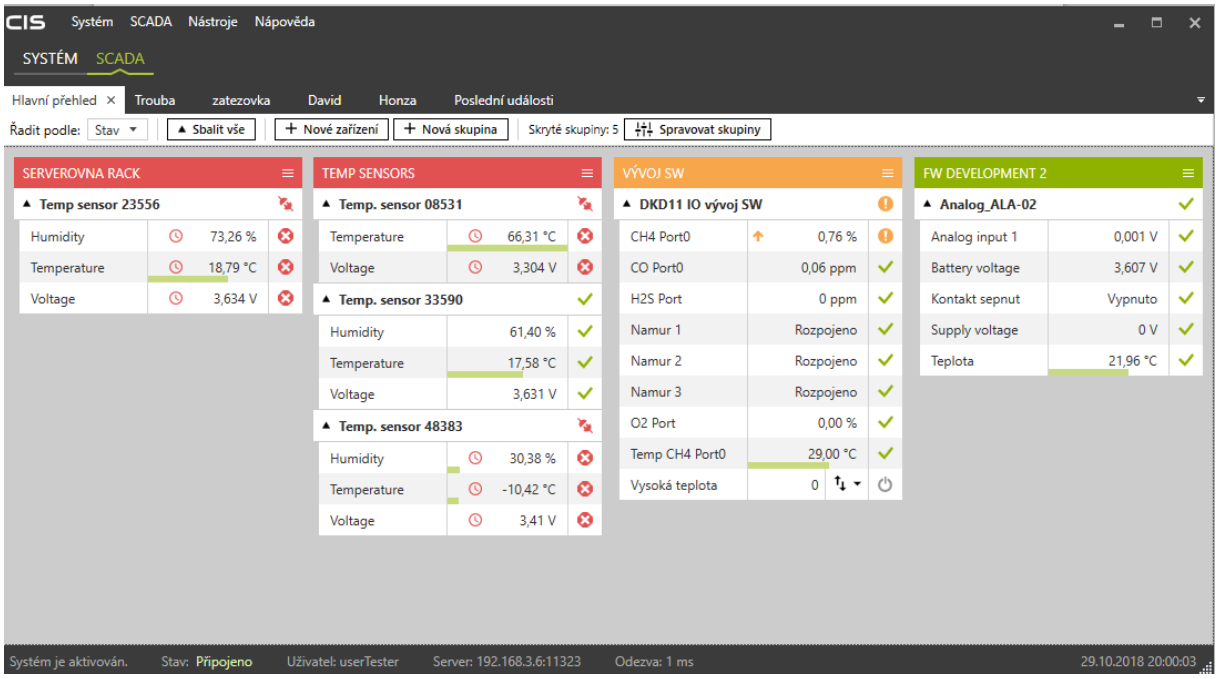
- **Nový hlavní přehled** - přidá na pracovní plochu záložku *Hlavní přehled*, viz kapitola 5.4 (tato záložka může být na pracovní ploše pouze jedenkrát),
- **Nové okno událostí** - přidá na pracovní plochu záložku *Okno událostí*, viz kapitola 5.6 (tato záložka může být na pracovní ploše pouze jedenkrát),
- **Nový vlastní pohled** - přidá na pracovní plochu záložku vlastního pohledu, viz kapitola 5.5 (tuto záložku lze přidávat bez omezení),
- **Odeslat rozložení pracovní plochy na server** - uloží aktuální rozložení pracovní plochy včetně všech nastavení na server,

- **Obnovit rozložení pracovní plochy** - načte aktuálně uložené rozložení pracovní plochy ze serveru a nahradí tak aktuální rozložení (pokud se liší).

V případě, že jiný aktuálně připojený uživatel provede změnu v rozložení pracovní plochy a odešle ji na server, jsou ostatní připojení uživatelé informováni a jsou dotázáni, zda si přejí nové rozložení načíst. V případě, že zvolí možnost *Ano*, je nové rozložení ihned načteno. V opačném případě zůstane zobrazeno původní rozložení do následujícího restartu CIS Klienta, případně do manuálního načtení rozložení ze serveru pomocí položky menu.

5.4 Hlavní přehled

Hlavní přehled je základní komponenta obsahující přehled všech aktuálně připojených zařízení rozdělených do skupin, připojených senzorů, případně nastavených výstupních pravidel. Základním prvkem přehledu je skupina, která je zobrazena jako stromová struktura obsahující přiřazená zařízení. Příklad takové struktury je zobrazen v obrázku 12.



Obrázek 12: Hlavní přehled SCADA modulu

Stavy koncových uzlů, tedy senzorů nebo výstupních pravidel, jsou propagovány směrem ke kořenu stromu tak, aby uživatel systému jednoduše identifikoval problém v rámci definované skupiny. Mezi tyto stavy patří výpadek příjmu dat senzoru nebo výstupu (rozdělen na dvě kategorie - varovná a kritická prodleva), dále dosažení varovných nebo kritických mezí a výpadek komunikace se zařízením. Každý ze stavu má svoji ikonu.

Horní lišta přehledu poskytuje uživateli možnost setřídit skupiny podle jejich stavu, jména, případně podle počtu zařízení. Zároveň poskytuje uživateli akce pro správu pracovní plochy,

které jsou popsány v samostatných podkapitolách. Po zvolení jakéhokoliv prvku ve stromové struktuře je ve spodní části přehledu zobrazen panel obsahující dostupné akce pro daný prvek.

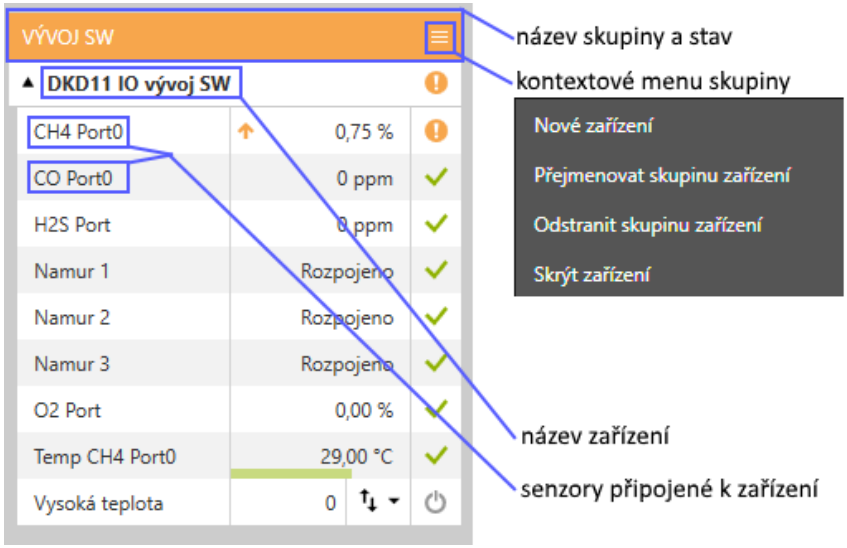
Ve výchozím nastavení pracovní plochy je tato komponenta zobrazena automaticky bez nutnosti dalšího nastavování a přispívá tak k rychlému zprovoznění celého systému.

5.4.1 Skupina

Definovaná zařízení je možné pro větší přehlednost zařadit do samostatných skupin (například podle umístění), které může uživatel s příslušným oprávněním spravovat. Zařízení, které nemá přiřazenou žádnou skupinu, je automaticky zobrazeno v automaticky vytvořené skupině **Nepřiřazená zařízení**.

Pro přidání nové skupiny slouží tlačítko *+Nová skupina* v horní liště hlavního přehledu. Zobrazí se dialogové okno pro definici nové skupiny. Přidání je možné dokončit stiskem tlačítka *Ok*, pro přidání další skupiny slouží tlačítko *Ok a další*.

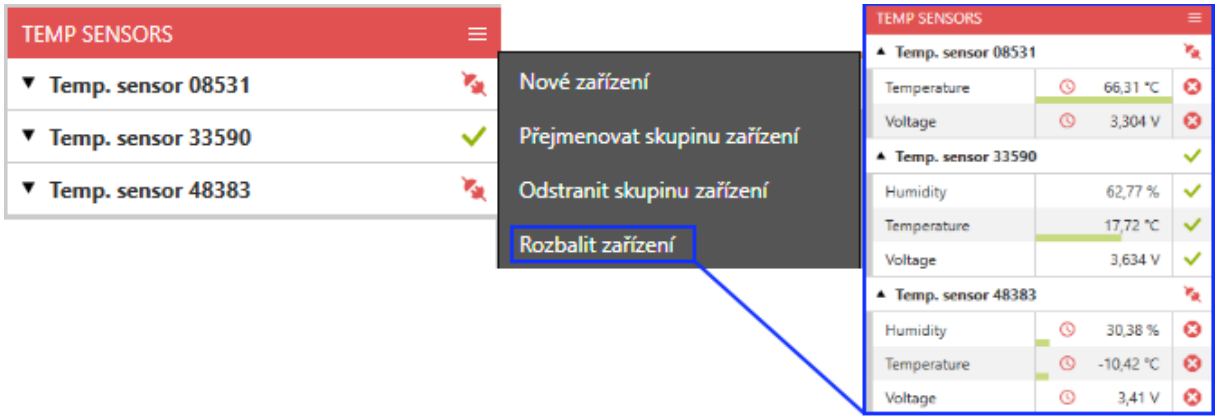
Editaci existující skupiny je možné provést dvojklikem na danou skupinu, případně vyvoláním menu pomocí ikony v pravém horním rohu okna skupiny. V tomto menu je také možné přidat nové zařízení, odstranit skupinu, nebo skrýt všechna zařízení ve skupině. Pro skrytí/zobrazení všech zařízení napříč všemi skupinami použijeme tlačítko *Skrýt vše/Zobrazit vše* v horní liště hlavního přehledu.



Obrázek 13: Náhled skupiny zařízení

5.4.2 Zařízení

Zařízení je v rámci skupiny zobrazeno tučným písmem. Pod ním jsou zobrazeny definované senzory a výstupy, které je možné libovolně skrývat nebo zobrazovat kliknutím na černou šipku vlevo od názvu zařízení. Ukázka obou stavů je zobrazena v obrázku č. 14.



Obrázek 14: Skupina v režimu sbaleno/rozbaleno

Editaci zařízení je možné provést dvojklikem na dané zařízení, případně vyvoláním menu pravým kliknutím myši na zařízení, kde je kromě editace možné zařízení odstranit. V případě editace se zobrazí okno s definicí zařízení, kde v horní části jsou zobrazeny základní informace, pod nimi následují tabulky s definicí senzorů a pravidel výstupů. Editaci je možné dokončit stiskem tlačítka *Potvrdit*, které uloží změny bez zavření okna, tlačítko *Ok* změny provede a okno následně uzavře. Správa senzorů a pravidel výstupů je popsána v samostatných kapitolách. Pomocí tlačítka *Nahrát pravidla* lze do zařízení nahrávat pravidla pro ovládání výstupů i v případě výpadku komunikace se serverem. Toto tlačítko je dostupné pouze u typů zařízení, které toto chování umožňují. Více je toto chování popsáno v kapitole 5.7.1. Příklad editačního dialogu zařízení DKD11 je zobrazen v obrázku č. 15.

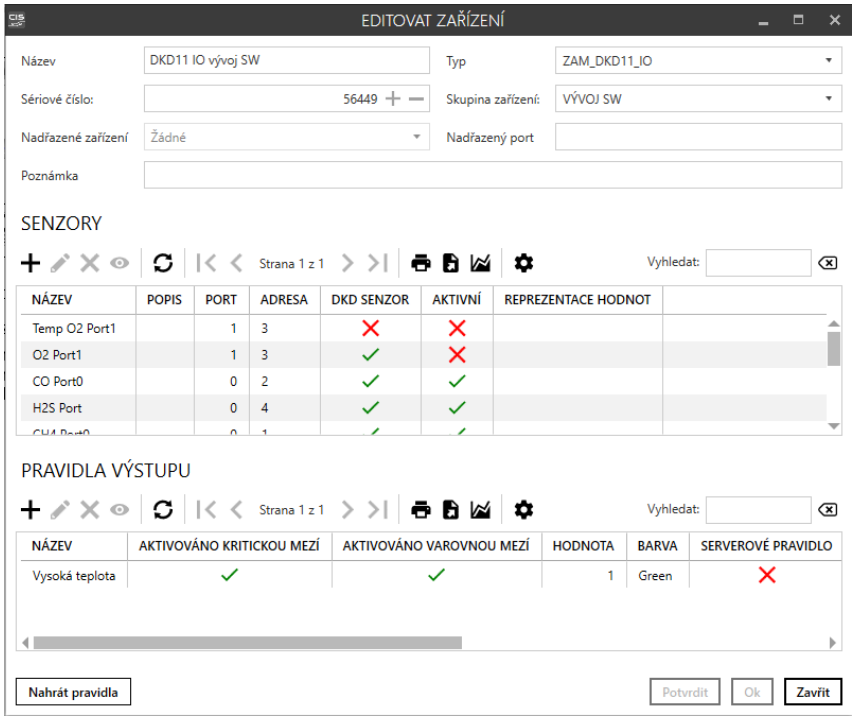
5.4.3 Senzor

Jedním z typů koncových uzlů stromové struktury skupiny zařízení jsou senzory. V poli senzoru je zobrazen samotný název, naměřená hodnota dle zadané veličiny, jednoduchý graf (pokud jsou nastaveny krajní meze veličiny) a stav na základě nastavených varovných/kritických mezí. V případě, že systém po definovanou dobu nepřijímá žádná data ze senzoru, je vedle jména zobrazena ikona budíku.

Senzor je možné editovat dvojklikem na pole senzoru. Kliknutím pravým tlačítkem myši na pole senzoru dojde k vyvolání menu, kde je možné senzor editovat, odstranit, případně zobrazit graf nebo protokol.

5.4.4 Pravidlo výstupu

Druhým typem koncového uzlu jsou pravidla výstupu, která na základě definovaných pravidel mění hodnotu na výstupu. Aktuálně jsou v systému uvažovány pouze binární výstupy. Obdobně jako u senzoru je v poli zobrazen název, aktuální hodnota, režim pravidla a stav. V případě, že systém po definovanou dobu nedostal informaci o hodnotě na výstupu, je vedle jména zobra-



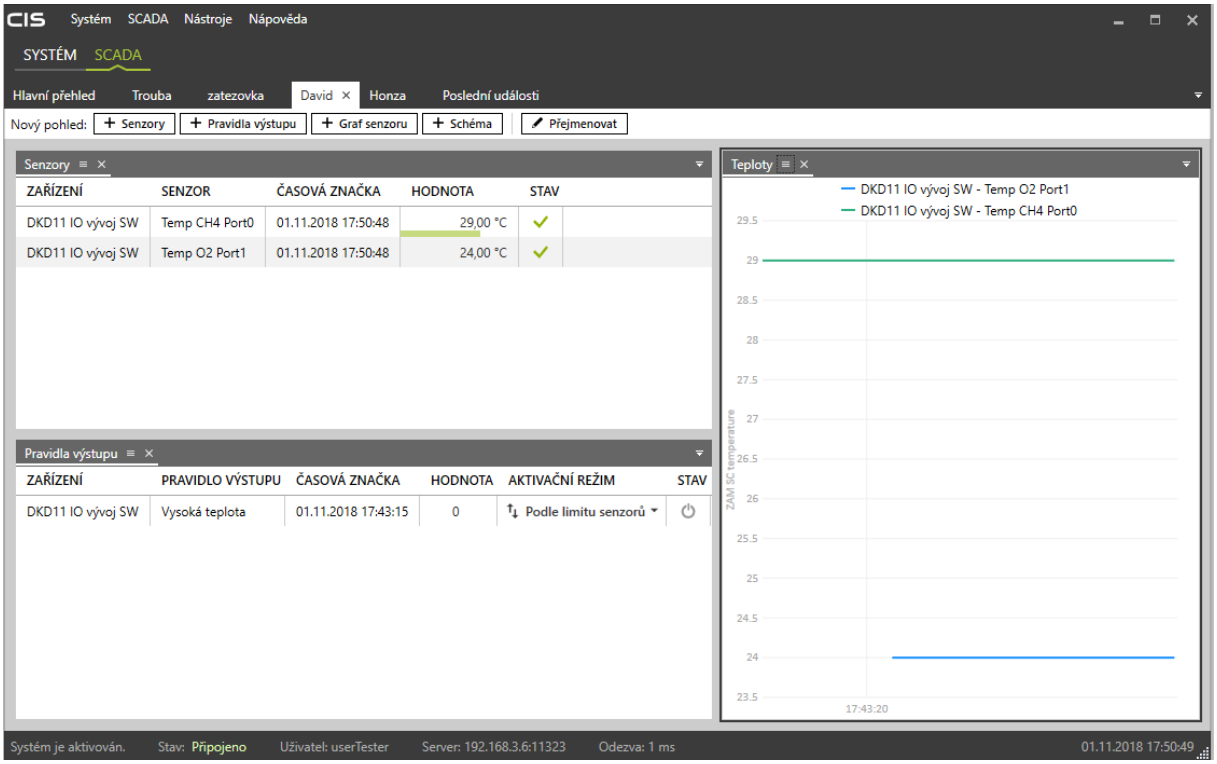
Obrázek 15: Pohled na editaci zařízení

zena ikona budíku. Režim pravidla je možné změnit kliknutím na ikonu režimu, na výběr jsou tři možnosti: **Podle limit senzorů** - nastavení hodnoty na výstupu je vyhodnoceno na základě definovaných pravidel, **Vynutit zapnutí** nastaví hodnotu, která by se standardně na výstup nastavila v případě splnění pravidla, **Vynutit vypnutí** naopak nastaví opačnou hodnotu.

Pravidlo výstupu je možné editovat dvojklikem na pole pravidla. Kliknutím pravým tlačítkem myši na pole pravidla dojde k vyvolání menu, kde je možné kromě editace pravidlo odstranit.

5.5 Vlastní pohled

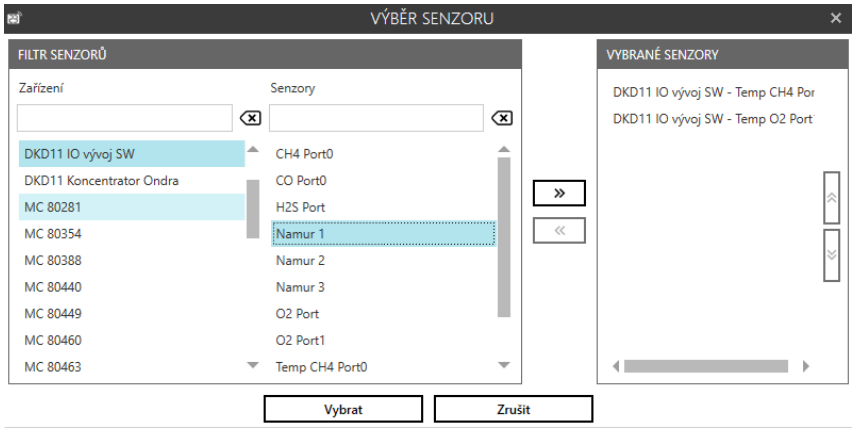
Vlastní pohled je rozšiřující komponenta, která je určena pro uživatele, kteří si chtějí pracovní plochu více přizpůsobit. Zvláště u instalací s větším počtem připojených zařízení (a tedy i senzorů a výstupů) je tak možné přehledně roztřídit sledované veličiny do samostatných komponent. V rámci této komponenty je možné vytvořit jeden z několika typů pohledů od základních tabulkových přehledů přes online grafy až po grafické schéma. Ukázka pohledu s kombinací různých komponent je zobrazena v obrázku č. 16. Jednotlivé typy pohledů jsou popsány v následujících podkapitolách. Počet vlastních pohledů není omezen, nicméně je nutné brát v potaz dostupné systémové prostředky počítače, kde je systém využíván. Zejména online grafy a schémata s větším množstvím objektů mohou způsobit větší využití operační paměti.



Obrázek 16: Ukázka vlastního pohledu

5.5.1 Pohled Senzory

Tabulkový pohled senzory umožňuje uživateli seskupit více senzorů nezávisle na jejich zařazení ve skupině nebo zařízení, ke kterému jsou připojeny. Je tak možné například přehledně zobrazit všechny senzory měřící stejnou veličinu v jedné tabulce. Zobrazené informace jsou totožné s těmi v hlavním přehledu. Přidání pohledu slouží tlačítko *+Senzory* v horní liště vlastního pohledu. Uživatel je následně vyzván k zadání jména a po potvrzení k výběru senzorů ze seznamu všech definovaných zařízení.



Obrázek 17: Okno pro výběr senzoru

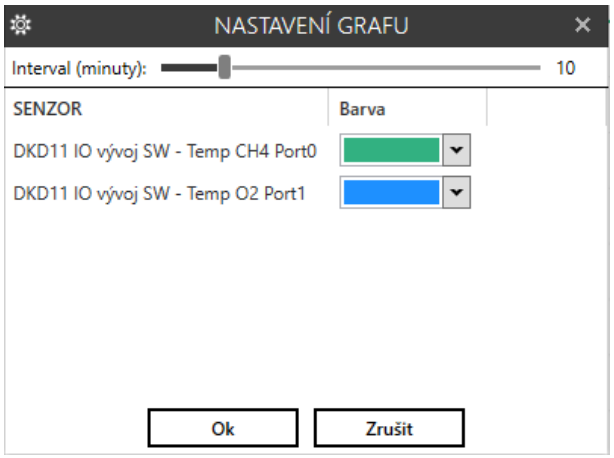
Okno pro výběr senzorů je rozděleno na dvě části. V levé je možné vyhledat požadovaný senzor v seznamu zařízení, v pravé části jsou uvedeny již vybrané senzory. Mezi oběma částmi je možné senzory přesouvat dvojklikem, případně za pomoci tlačítek » a «.

5.5.2 Pohled Pravidla výstupu

Druhý tabulkový pohled na pravidla výstupů poskytuje obdobnou funkcionalitu jako tabulkový pohled na senzory. Uživatel si může přehledně seskupit pravidla výstupů napříč vícero zařízeními, příkladem může být ovládání světel na pracovišti. Pro přidání pohledu slouží tlačítko *+Pravidla výstupu* v horní liště vlastního pohledu.

5.5.3 Pohled Graf senzoru

Grafová komponenta poskytuje online přehled o stavu vybraných veličin, které mohou pocházet z vícero zařízení. Přidání je možné kliknutím na tlačítko *+Graf senzoru* v horní liště vlastního pohledu. Začátek nastavení probíhá obdobně jako u tabulkových komponent, následně se uživateli zobrazí okno nastavení grafu. V něm je možné nastavit barvu jednotlivých senzorů a časový úsek, který se má zobrazovat.



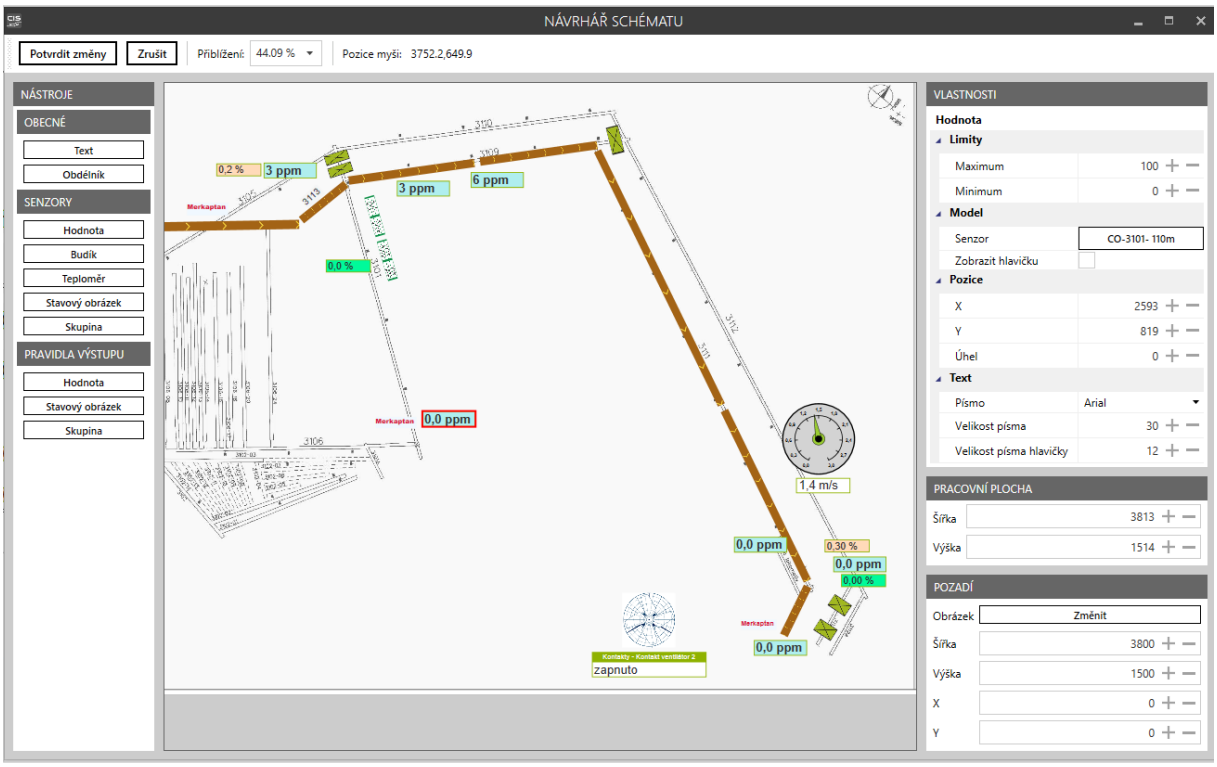
Obrázek 18: Okno nastavení grafu

5.5.4 Pohled Schéma

Schéma je nejpokročilejší komponenta vlastního pohledu a umožňuje uživateli vizualizovat monitorované pracoviště. Je tedy například možné do schématu umístit plánek budovy a s využitím dostupných grafických komponent zobrazit měřené veličiny v místech, kde probíhá měření. Pro přidání nového schématu slouží tlačítko *+Schéma* v horní liště vlastního pohledu. Spustí se okno s návrhářem schématu, kde probíhá veškeré nastavení. Toto okno je rozděleno na tři části:

- **nástroje** - obsahují seznam dostupných komponent rozdělený do tří kategorií

  - **obecné** - komponenty se statickým obsahem
  - **senzory** - komponenty pro vizualizaci senzorů
  - **pravidla výstupu**- komponenty pro vizualizaci pravidel výstupů
- **pracovní plocha** - obsahuje samotné schéma
- **nastavení** - obsahuje vlastnosti aktuálně zvolené komponenty, nastavení pracovní plochy a nastavení pozadí



Obrázek 19: Ukázka návrháře schématu

Přidání nové komponenty je možné provést kliknutím na požadovaný typ v panelu nástrojů. Ten se následně objeví v levém horním rohu pracovní plochy, kde je s ním možné dále pracovat. Po kliknutí na umístěnou komponentu se v panelu vlastnosti zobrazí dostupná nastavení rozdělená do kategorií (některé jsou společné pro všechny komponenty, jako pozice a velikost, jiná jsou specifická pro daný typ komponenty). Nejvíce přizpůsobitelnou komponentou je *stavový obrázek* dostupný pro vizualizaci senzoru i pravidla výstupu. Jak už název napovídá, tato komponenta zobrazuje obrázek podle aktuální hodnoty senzoru nebo výstupu na základě definovaných pravidel, viz kapitola 5.7.4. Kromě nastavení samotných komponent je možné konfigurovat velikost pracovní plochy a nastavit pozadí.

Dokončení nastavení provedeme kliknutím na tlačítko *Provést změny*. Přiblížení nebo oddálení schématu je možné podržením klávesy *Ctrl* a otáčením kolečkem myši.

5.6 Okno událostí

Okno událostí je informativní komponenta sloužící převážně pro správce systému. Zobrazuje historií události, které nastávají při komunikaci se zařízeními. Příkladem je informováním o stavu automatického nahrávání pravidel do zařízení *DKD11* a *DKDIO* po změně definic senzorů, případně pravidel výstupů. Další informace o nahrávání pravidel jsou dostupné v kapitole 5.7.1.

5.7 Administrace

Tato kapitola popisuje správu všech definic ve SCADA modulu. Přohlížet a spravovat definice je možné v menu **Administrace** v hlavním menu modulu, některé definice je zároveň možné spravovat přímo z pracovní plochy. Administrace je dostupná uživatelům s oprávněním **Administrace**. Definice jsou rozděleny do dvou kategorií:

- **definice typů** - ovlivňují chování celého systému (definice veličin, typů zařízení, výstupů zařízení), jsou předdefinované při instalaci,
- **instance typů** - neovlivní chování celého systému (definice zařízení, připojených senzorů, pravidel výstupů), jsou definovány uživatelem.

Všechny položky administrace zobrazí samostatné okno s tabulkovou komponentou, některé jsou typu *master-detail*, kde v horní části jsou aktuálně spravovaná data, v dolní části jsou tabulky s daty, které mají na spravovaná data vazbu.

5.7.1 Typy zařízení

V tomto okně je možné zobrazit přehled všech typů zařízení definovaných v systému, které je možné používat v modulu SCADA. Tyto typy jsou předdefinovány při instalaci a není možné je uživatelsky editovat.

Kromě samotného názvu typu je zde zobrazena informace, zda zařízení umí v případě výpadku komunikace se serverem autonomně nastavovat hodnoty na výstupech na základě definovaných pravidel (s tím omezením, že pravidla se mohou vázat pouze na senzory připojené k tomuto zařízení). U zařízení, které tuto funkci podporují, jsou automaticky v pravidelných časových intervalech kontrolovány změny nastavení. V případě detekce změny se systém pokusí definovaná pravidla nahrát do zařízení. Nahrání pravidel je také možné vyvolat ručně v editačním dialogu konkrétního zařízení.

Dále je u každého typu zařízení zobrazena informace, zda existuje jako rozšíření jiného zařízení a nemůže samostatně existovat. Ve spodní části okna jsou zobrazeny typy výstupů pro vybrané zařízení. Jejich editace je popsána v kapitole 5.7.2.

5.7.2 Výstupy

Tabulka výstupů zobrazuje definice typů výstupů pro konkrétní zařízení. Výstup je obdoba vstupu (senzoru) s tím rozdílem, že jeho hodnotu je možné ovlivnit ze systému ručně nebo

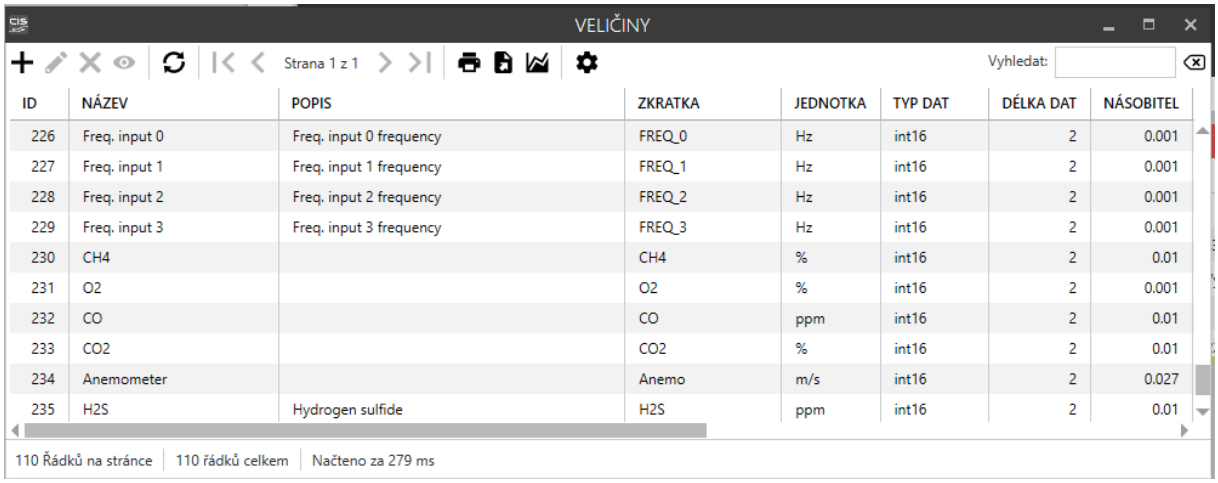
na základě definovaných pravidel. Oproti senzorům nejsou výstupy univerzální a vážou se na konkrétní typ zařízení. Systém byl navržen s ohledem na více druhů výstupů (binární, vícestavové), aktuálně je ale pouze implementována binární varianta.

V rámci zařízení je výstup identifikován číslem portu, případně sekundární adresou v rámci daného portu (například adresa v rámci sběrnice *Modbus*).

5.7.3 Veličiny

Tabulka veličin zobrazuje definice typů vstupů (senzorů), které systém umí zpracovávat a správně zobrazit. Veličiny nejsou vázány na konkrétní zařízení a je možné je libovolně editovat. Níže je uveden popis jednotlivých parametrů veličiny, které je možné uživatelsky nastavit. Ukázka definic veličin je zobrazena v obrázku č. 20.

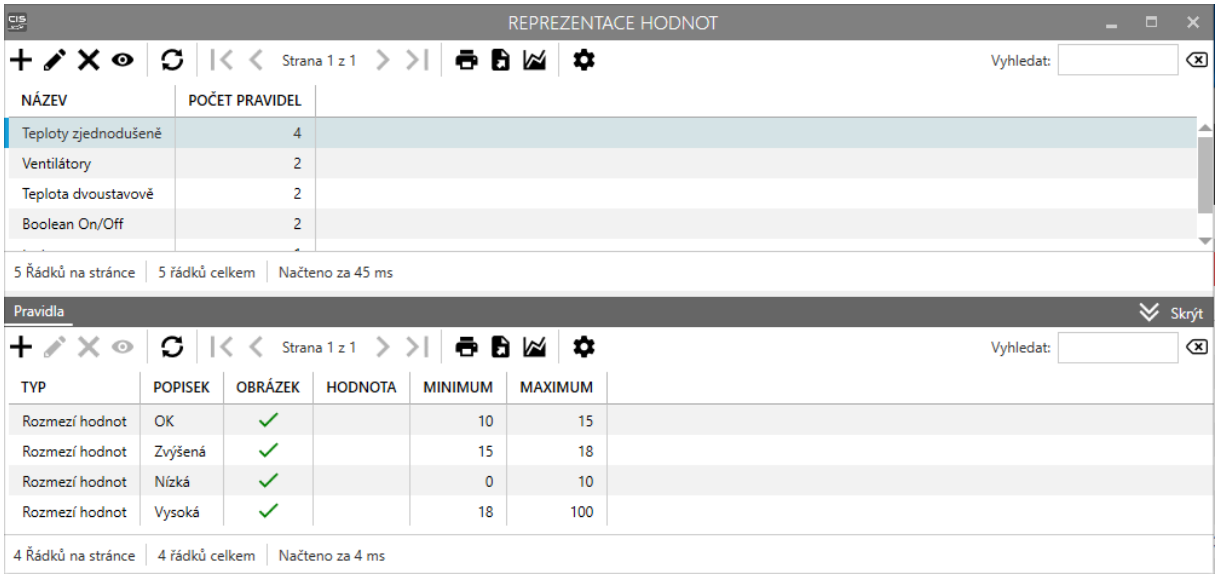
- **ID** - unikátní identifikátor veličiny, který slouží pro identifikaci naměřených dat ze zařízení,
- **Jméno** - název veličiny v systému,
- **Popis** - bližší informace o veličině,
- **Zkratka** - zkrácený název veličiny,
- **Jednotka** - fyzikální jednotka veličiny,
- **Typ dat** - určuje, v jakém datovém typu jsou data uložena,
- **Délka dat** - určuje počet bytů, ve kterých jsou data v rámci datového typu uložena,
- **Počet des. míst** - určuje počet desetinných míst,
- **Pojmenování hodnot** - výchozí nastavení reprezentace hodnot, viz kapitola 5.7.4,
- **Násobitel** a **Konstanta** - hodnoty pro přepočet v lineární rovnici $y = a * x + b$,
- **Kumulativní** - určuje, zda je veličina kumulativní, stále rostoucí hodnota,
- **Kumulativní interval** - pokud je veličina kumulativní, interval představuje období, po které je vypočítáván přírůstek,
- **Limity** - limity hodnot, kterých může veličina nabývat,
- **Kritické/varovné meze** - výchozí meze hodnot pro danou veličinu.



Obrázek 20: Ukázka definic veličin

5.7.4 Pojmenování hodnot

Pojmenování hodnot je funkce, která vyplynula ze zkušeností získaných během pilotního testování systému. Umožňuje uživateli zobrazit místo naměřené hodnoty libovolný text, který se může měnit na základě pravidel definovaných u dané reprezentace. Okno je rozděleno na dvě části, v horní části je zobrazen přehled aktuálně definovaných pojmenování, ve spodní části pak pravidla v rámci zvoleného pojmenování. Pravidla pojmenování je možné nastavit jak pro konkrétní hodnotu, tak i pro rozsah hodnot. Hodnoty mimo rozsahy definovaných pravidel jsou zobrazovány standardně bez pojmenování. Ke každému pravidlu je také kromě popisku možné nastavit obrázek, který je zobrazován ve schématu v rámci komponenty *Stavový obrázek*. Příklad definice pojmenování včetně pravidel je zobrazen v obrázku č. 21.



Obrázek 21: Ukázka definice pojmenování hodnot

Jednotlivá pojmenování je následně možné přiřadit k definici senzoru nebo pravidlu výstupu, případně i k definici veličiny. V posledním případě se pravidly pro pojmenování bude následně řídit každý senzor měřící tuto veličinu, pokud sám nemá přiřazené jiné pojmenování.

5.7.5 Zařízení

V tabulce zařízení je přehled všech aktuálně definovaných zařízení v systému. Zobrazené informace i dostupné akce jsou totožné s těmi v hlavním přehledu, kterým se zabývá kapitola 5.4.2. Ve spodní části jsou zobrazeny přehledy senzorů a pravidel výstupů pro vybrané zařízení.

5.7.6 Senzory

Tabulka senzorů zobrazuje přehled všech definovaných senzorů napříč všemi zařízeními. Část parametrů senzorů vychází z parametrů veličiny, kterou senzor měří. Nově přidaný senzor tak například získá výchozí nastavení kritických a varovných mezi nebo reprezentaci hodnot, které je ale možné následně upravit. Spravovat senzory v rámci zařízení je možné i přímo z hlavního přehledu. Níže je uveden přehled parametrů senzoru>

- **Název** - pojmenování senzoru,
- **Aktivní** - viditelnost senzoru v přehledech, oproti trvalému odstranění senzoru může sloužit například pro dočasné vyřazení,
- **Veličina** - nastavení veličiny, kterou senzor měří,
- **Zařízení** - výběr zařízení, ke kterému je senzor připojen,
- **Popis** - doplňující popis senzoru,
- **DKD senzor** - v případě připojení senzoru k důlnímu koncentrátoru tento parametr značí, zda se na něj mohou vztahovat autonomně vyhodnocovaná pravidla v rámci zařízení, viz kapitola 5.7.1,
- **Port** - pokud není senzor přímo součástí zařízení, značí parametr port, ke kterému je senzor připojen,
- **Adresa** - adresa v rámci nastaveného portu,
- **Pojmenování hodnot** - nastavení reprezentace hodnot, pokud není nastaveno, je použito nastavení reprezentace veličiny,
- **Komunikace** - představuje dobu, po které je vyhlášena výstraha v případě výpadku komunikace,
- **Kritické/varovné meze** - meze hodnot pro danou veličinu, výchozí nastavení je převzato z vybrané veličiny.

5.7.7 Pravidla výstupů

Tabulka pravidel výstupů zobrazuje přehled všech definovaných pravidel výstupů. Popis parametrů pravidel výstupů je uveden níže:

- **Zařízení** - výběr zařízení, ke kterému se pravidlo váže,
- **Název** - pojmenování pravidla,
- **Aktivní** - viditelnost pravidla v přehledech, funguje obdobně jako nastavení aktivity u senzoru,
- **Výstup** - výstup v rámci vybraného zařízení, který bude pravidlem ovládán,
- **Aktivováno varovnou/kritickou mezí** - značí, zda bude pravidlo sepnuto po dosažení kritické/varovné meze u vybraných senzorů,
- **Hodnota** - nastavení hodnoty, která bude nastavena na výstup v případě aktivace pravidla (pouze 0 nebo 1),
- **Barva** - nastavení barvy, která bude zobrazena v hlavním přehledu při aktivaci pravidla,
- **Reprezentace hodnot** - nastavení reprezentace hodnot,
- **Serverové pravidlo** - určuje, zda je pravidlo možné vyhodnocovat autonomně v rámci zařízení, tento parametr je dopočítán automaticky na základě ostatních nastavení a není možné ho uživatelsky měnit,
- **Senzory** - senzory, které ovlivní chování pravidla (výběr probíhá v samostatném okně, které je totožné s oknem pro výběr senzoru do vlastního pohledu, viz obrázek č. 17).

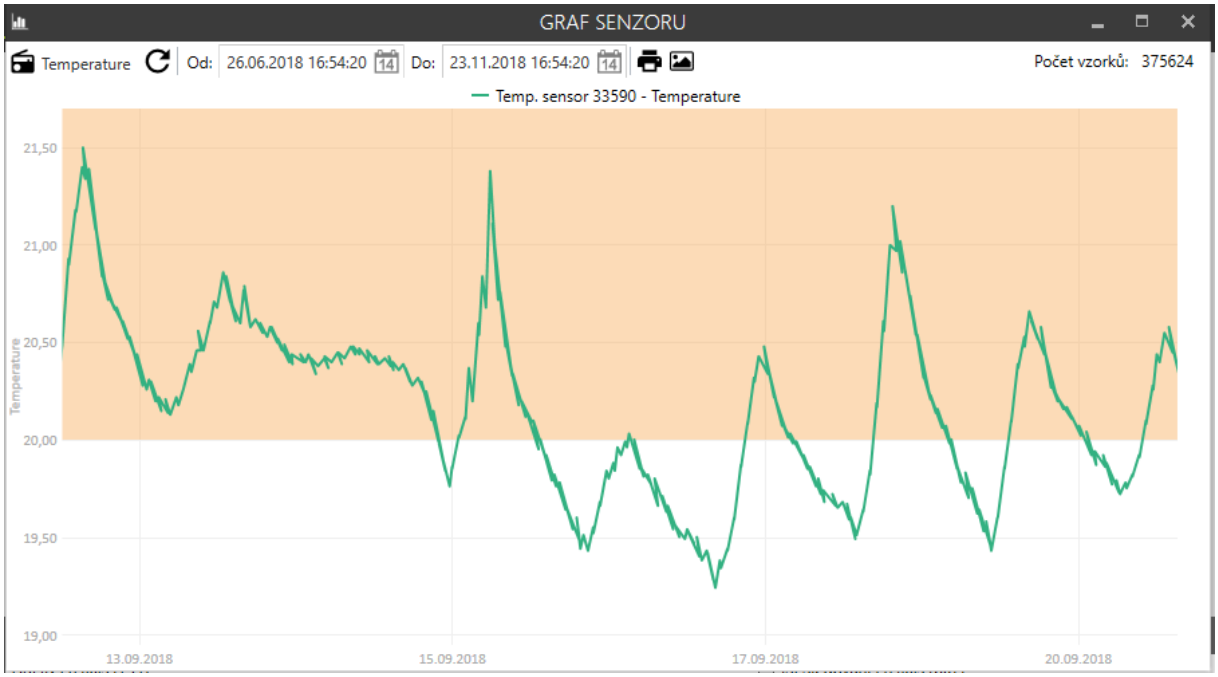
5.8 Práce s naměřenými daty

Kromě online přehledů je možné s naměřenými daty pracovat několika způsoby, které jsou popsány v této kapitole. Systém poskytuje funkce umožňující jednak rychlou kontrolu v rámci systému, ale i vytváření souhrnných reportů, případně přípravu dat pro zpracování v jiných systémech.

5.8.1 Grafy

Funkce grafu je dostupná v menu *SCADA*, podmenu *Grafy*. Zobrazí se nové okno, kde v horní liště jsou zobrazeny všechny dostupné akce. Mezi základní patří výběr požadovaného senzoru (v samostatném okně, které již bylo popisováno v dřívějších kapitolách), výběr časového úseku, ze kterého mají být data vizualizována. Nechybí ani možnost vytvoření obrázku nebo tisk. Ukázku grafu je možné nalézt v obrázku č. 22. Osa X znázorňuje čas, osa Y naměřená data.

Pokud má vybraný senzor nastavené varovné nebo kritické meze, jsou v grafu vyznačeny odpovídající barvou. Přiblížit nebo oddálit zobrazení v obou osách je možné otáčením kolečka myši v prostoru grafu. V případě potřeby je možné přiblížit nebo oddálit pouze jednu osu otáčením kolečka myši v prostoru požadované osy.

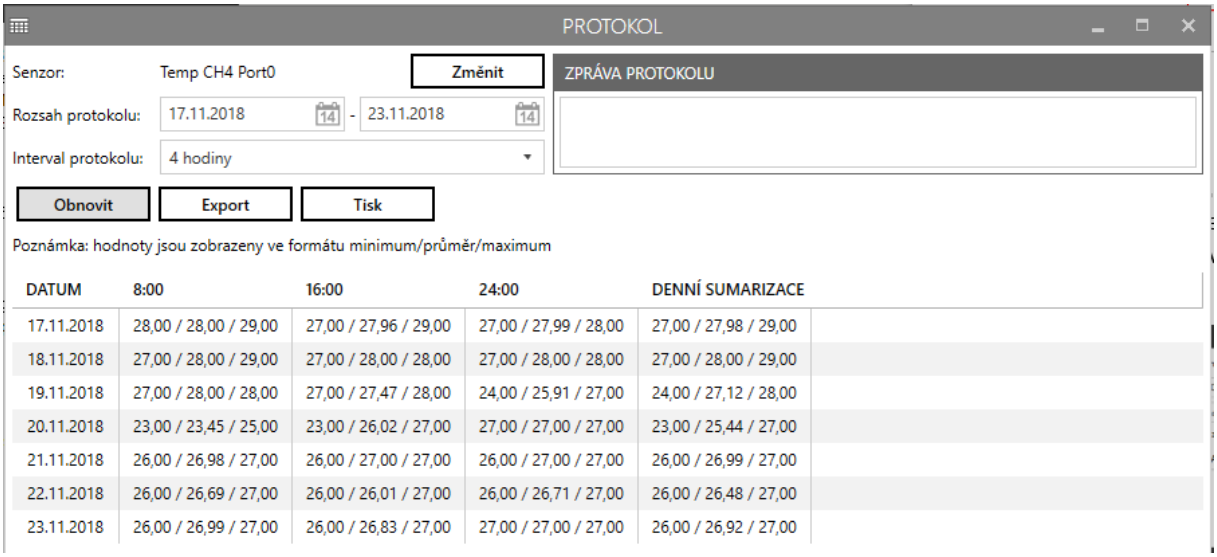


Obrázek 22: Ukázka vizualizace dat pomocí grafu

5.8.2 Protokoly

Pomocí funkce protokolu je možné získat sumarizaci naměřených dat u zvoleného senzoru za zvolené období. Tato funkce je dostupná v menu *SCADA*, podmenu *Protokol*. V zobrazeném okně je v horní části dostupné nastavení kde je možné zvolit požadovaný senzor, časový úsek a interval protokolu, ve kterém budou prováděny sumarizace.

Samotný protokol je zobrazen v tabulce, kde řádky představují dny vybraného časového úseku, ve sloupcích jsou uvedeny sumarizace ve zvoleném intervalu. V posledním sloupci jsou uvedeny sumarizace za celý den. Protokol je možné vytisknout (v případě, že byla vyplněna zpráva protokolu v pravé horní části okna, bude tato zpráva v tisku zahrnuta). Zároveň je možné protokol exportovat ve formátu CSV nebo XLSX. Ukázku protokolu je možné nalézt na obrázku č. 23.



DATUM	8:00	16:00	24:00	DENNÍ SUMARIZACE
17.11.2018	28,00 / 28,00 / 29,00	27,00 / 27,96 / 29,00	27,00 / 27,99 / 28,00	27,00 / 27,98 / 29,00
18.11.2018	27,00 / 28,00 / 29,00	27,00 / 28,00 / 28,00	27,00 / 28,00 / 28,00	27,00 / 28,00 / 29,00
19.11.2018	27,00 / 28,00 / 28,00	27,00 / 27,47 / 28,00	24,00 / 25,91 / 27,00	24,00 / 27,12 / 28,00
20.11.2018	23,00 / 23,45 / 25,00	23,00 / 26,02 / 27,00	27,00 / 27,00 / 27,00	23,00 / 25,44 / 27,00
21.11.2018	26,00 / 26,98 / 27,00	26,00 / 27,00 / 27,00	26,00 / 27,00 / 27,00	26,00 / 26,99 / 27,00
22.11.2018	26,00 / 26,69 / 27,00	26,00 / 26,01 / 27,00	26,00 / 26,71 / 27,00	26,00 / 26,48 / 27,00
23.11.2018	26,00 / 26,99 / 27,00	26,00 / 26,83 / 27,00	27,00 / 27,00 / 27,00	26,00 / 26,92 / 27,00

Obrázek 23: Ukázka protokolu

5.8.3 Export dat

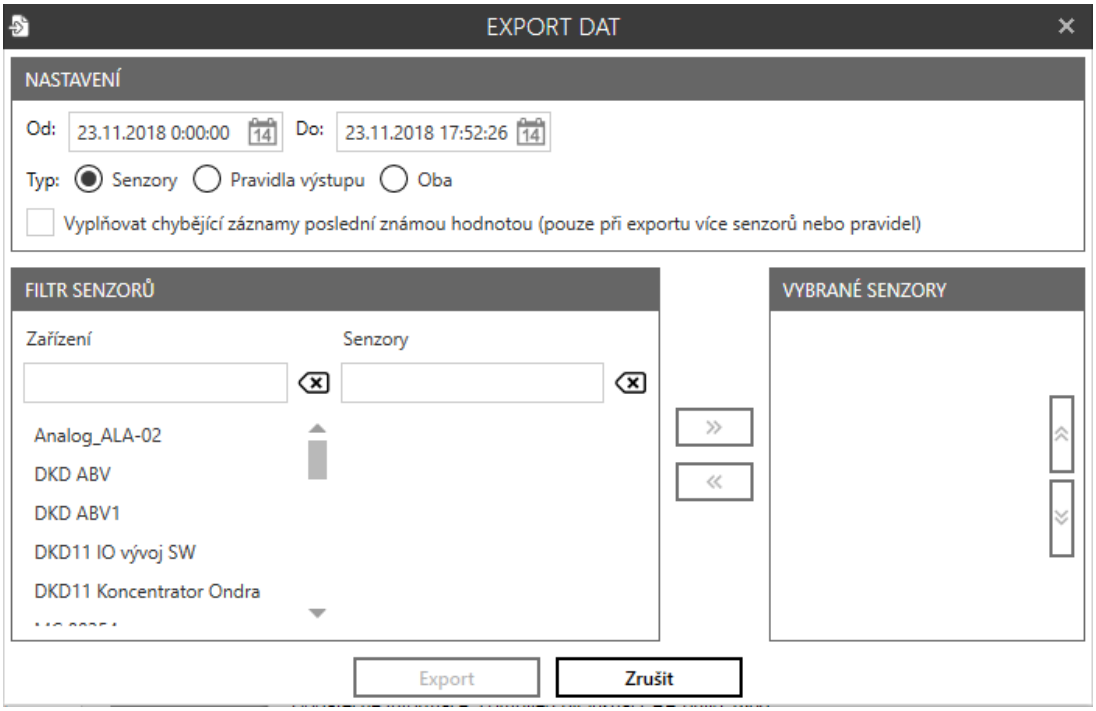
V případě potřeby je možné naměřená data senzorů a výstupů (dále jen datové zdroje) exportovat ve formátu CSV. Tato funkce je dostupná v menu *SCADA*, podmenu *Export dat*. Zobrazí se okno nastavení exportu, v horní části je dostupné základní nastavení, ve spodní části je pak možné vybírat požadované datové zdroje. Jednotlivé položky nastavení jsou přiblíženy níže:

- **časový interval** - v tomto časovém intervalu budou data exportována,
- **typ** - určuje, zda bude export složen z dat senzorů, výstupů, případně kombinace obojího (nastavení ovlivní spodní část okna, kde je možné vybírat datové zdroje),
- **vyplňování chybějících záznamů** - je přiblíženo v popisu exportovaného souboru.

V prvním sloupci exportovaného souboru je uvedena časová značka naměřených dat s přesností na sekundy (tzn. pokud kterýkoliv z datových zdrojů v daném čase naměřil data, je vytvořen nový řádek). V následujících sloupcích jsou uvedena data z vybraných zdrojů. Často se stává, že některý zdroj pro danou časovou značku nemá dostupná data, vyplnění buňky se v tomto případě řídí nastavením *vyplňování chybějících záznamů*. V případě zaškrtnutí této možnosti bude buňka vyplněna předchozí hodnotou datového zdroje. V opačném případě zůstane buňka prázdná.

5.9 Servisní záznamy

Servisní záznamy jsou doplňkovou funkcí, která umožňuje uživateli vytvořit událost zdůvodňující výpadek komunikace zařízení, například z důvodu servisního zásahu, odstávky, případně výměny.



Obrázek 24: Export dat

Mohou tak například vysvětlit mezeru v protokolu při revizní kontrole za delší časový úsek. Záznamy mají pouze informativní charakter a nijak neovlivňují chování zbytku systému.

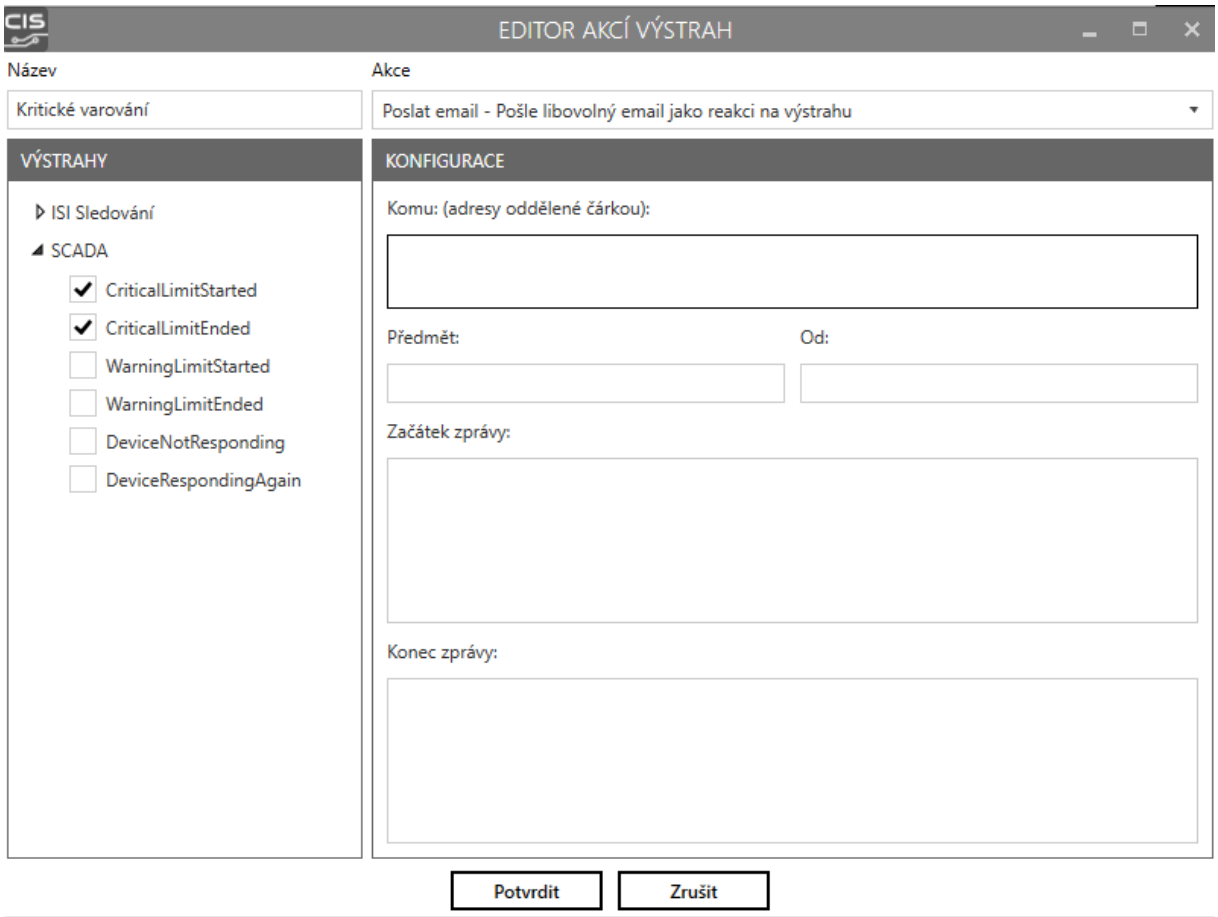
Správa servisních záznamů je dostupná z menu *SCADA*, podmenu *Servisní záznamy*. Při přidání servisního záznamu je možné specifikovat pojmenování události, časový interval a komentář blíže popisující prováděné Akce. Kromě těchto obecných informací je také možné vybrat zařízení, kterých se událost týká.

5.10 Výstrahy

Scada využívá systém výstrah implementovaný v jádru systému. Uživatel je na nové výstrahy upozorněn automatickým vyvoláním notifikačního okna s počtem aktivních výstrah, zároveň je také přehráván výstražný zvuk. Detail výstrah je možné zobrazit kliknutím na toto okno, kde je v tabulce zobrazen čas, typ výstrahy, určení místa (senzor, zařízení, případně hodnota, která vyvolala událost) a stav. Zároveň je zde možné výstrahu akceptovat (vzít na vědomí). Scada informuje o následujících událostech:

- výpadek komunikace se zařízením,
- obnovení komunikace po výpadku,
- dosažení varovných/kritických mezí,
- návrat do normálních hodnot.

V menu nastavení výstrah je možné ke každému typu události nastavit reakci, například zaslání emailu nebo SMS správci systému. Ukázku nastavení je možné nalézt v obrázku č. 25.



Obrázek 25: Ukázka nastavení reakcí na výstrahu.

6 Instalace

V této části je popsána instalace CIS serveru. Jak již bylo uvedeno v předchozí kapitole, předpokládá se základní znalost práce s databází PostgreSQL. Pro sestavení aplikace je nutné mít nainstalované vývojové prostředí Visual Studio 2015 nebo novější. Konfigurační soubor CIS serveru je uložen jako SQLite databáze, pro jeho nastavení je tedy nutný libovolný prohlížeč SQLite databází.

1. Nejprve je nutné nainstalovat databázový systém PostgreSQL (9.5 nebo vyšší). V případě instalace na OS Windows je na webových stránkách dostupný instalátor obsahující samotný databázový systém spolu s aplikací pgAdmin4 - nástroj pro správu a vývoj databází[9]. V každém kroku je možné ponechat výchozí nastavení kromě nastavení uživatelského účtu, které je nutné znát pro další použití. Případně je možné využít již stávající instalaci, která je k dispozici.

2. V nainstalovaném databázovém systému je nutné vytvořit novou databázi a následně do ní obnovit předpřipravenou zálohu z přiloženého souboru **scada.backup** umístěného v adresáři **database**. To můžeme provést například za pomoci nástroje pgAdmin4, který je součástí instalačního balíčku pro Windows, viz předchozí krok.

3. Za pomoci scriptu **build.bat** v adresáři **src** sestavíme aplikaci. Sestavená aplikace se uloží do adresáře **src\Binaries\Server**. Alternativně lze aplikaci sestavit přímo z jednotlivých **sln** souborů ve vývojovém prostředí Visual Studio:

   - CIS Server z src\server\Server.sln,
   - CIS Client z src\client\Client.sln,
   - SCADA modul z src\modules\scada\Scada.sln.

4. Do adresáře se sestavenou aplikací zkopírujeme složku **Certificates** včetně jejího obsahu z adresáře **dist**. Do stejného adresáře zkopírujeme i výchozí konfigurační soubor **config.dat** z adresáře **dist**.

5. Za pomoci libovolného prohlížeče SQLite databází otevřeme soubor **config.dat** v adresáři sestavené aplikace. Konfigurační soubor obsahuje následující tabulky, jejichž záznamy můžeme upravovat:

   - **t_devices** - definuje komunikační kanály pro komunikaci se zařízeními, typické instalaci postačuje jeden otevřený komunikační kanál - tcp_server poslouchající na uvedeném portu
   - **t_values** - obsahuje konfigurační položky jádra systému a instalovaných modulů, nejdůležitější z nich jsou uvedeny níže:
     - **ListenerPort**: port, na kterém server přijímá spojení s klienty

- **LogLevel**: úroveň logování: Tosh(5), Debug(4), Info(3), Warning(2), Error(1), výchozí hodnota je Info(3)
- **DBHos**t: adresa databázovéhou serveru
- **DBPort**: port, na kterém databázový server přijímá spojení
- **Database**: název databáze (název databáze vytvořené v předchozím kroku)
- **DBUser**: uživatel vytvořený při instalaci databáze
- **DBPassword**: heslo uživatele zadané při instalaci databáze

6. Nyní je možné ověřit funkčnost spuštěním serveru jako konzolovou aplikaci - spuštěním souboru **ISExecutable.exe**

7. Pokud je vše funkční, můžeme nainstalovat server jako službu. V adresáři serveru otevřeme příkazový řádek a spustíme soubor **ISService.exe** s parametrem /install. Příkazový řádek je nutné spustit jako administrátor.

8. Tímto je instalace serverové části dokončena, můžeme se tedy zkusit přihlásit z klientské aplikace - postup je popsán v kapitole 5.1.2, instalační soubor je umístěn v adresáři **dist**. Alternativně lze překopírovat klientskou část ze sestavené aplikace z adresáře **src\Binaries\Server\Content\Update** Pro testovací účely byl ve výchozí databázi připraven uživatelský účet s údaji **scada/scada**.

7 Navazující práce

Vzhledem k ročnímu prodloužení termínu odevzdání práce je možné popsat následné úpravy a rozšíření, ke kterým došlo na základě zkušeností z nasazení v produkčním prostředí.

Přestože byl systém testován na firemním testovacím pracovišti s několika připojenými zařízeními, ne všechny chyby byly po nasazení do produkčního prostředí odladěny. Jedna taková chyba, jejíž identifikace a oprava zabrala několik desítek hodin času, nastávala po vytisknutí protokolu v případě spuštění aplikace na OS Windows 7. Po dokončení tisku se napříč celou aplikací přestávaly v textech zobrazovat některé speciální znaky. K nápravě došlo až po restartování aplikace. Tato chyba byla velmi obtížně reprodukovatelná, protože uživatel, který chybu nahlásil, nedokázal popsat scénář, který k výše popisovanému chování vedl a podezření nejprve padlo na chybné nastavení operačního systému nebo chybějící font. Nakonec byl problém identifikován ve funkci generující XPS dokument s protokolem, chyba byla v samotném WPF frameworku. Řešením byla změna způsobu generování dokumentu.

Jednou z navazujících úprav bylo omezení zobrazení skupin zařízení v hlavním přehledu pro jednotlivé uživatele. U instalace pracoviště rozděleného do několika oddělení totiž přišel požadavek, aby uživatelé z jednotlivých oddělení viděli pouze připojená zařízení v rámci jejich oddělení.

Rozšíření se také týkalo grafického schématu. Původní návrh nepočítal s komponentou stavového obrázku pro senzory a výstupní pravidla. Jak se ale ukázalo po nasazení do produkčního prostředí, dostupné komponenty nepostačovaly pro kompletní vizualizaci pracoviště a implementace stavového obrázku byla řešením, které nahradilo nutnost implementace jiných specifických komponent. Kreativnější uživatelé tak zároveň získali možnost přizpůsobit si pohled podle svých potřeb (typickým příkladem je vizualizace stavu pohyblivého pásu).

Následující úpravou, tentokrát na straně serveru, byla implementace komunikace s *PLC automaty AMiT*, u kterých měl systém být schopen vyčítat hodnoty na vstupech a zároveň měl ovládat nastavení hodnot na výstupech (chování tedy mělo být stejné jako u stávajících podporovaných zařízení). V rámci implementace tohoto požadavku bylo zpracování dat více zobecněno tak, aby bylo do budoucna možné snadno přidávat nová komunikační rozhraní.

Dalším rozšířením byl požadavek na implementaci ovládání osvětlení důlních chodeb s podporou funkce tzv. „světelné odvolávky“, tedy v případě požadavku dispečera mělo osvětlení v definovaných intervalech blikat. Pro tyto účely byly využity výstupy zařízení, kdy každý výstup ovládá určitou sekci světel. V rámci implementace byl vytvořen jednoduchý CIS modul, který s využitím rozhraní modulu SCADA (knihovna *ScadaApi*) výstupy ovládá.

Posledním významným rozšířením, které bylo implementováno, je poskytování naměřených dat pomocí protokolu *OPC* jiným systémům. OPC je komunikační standard pro bezpečnou a spolehlivou výměnu dat v prostředí průmyslové automatizace. Je platformově nezávislý a zajišťuje výměnu informací napříč zařízeními různých výrobců.[10] V tomto režimu může CIS

se SCADA modulem fungovat pouze jako podsystém zprostředkující komunikaci se zařízeními společnosti CUTTER Systems spol. s r.o.

8 Závěr

Primárním cílem práce bylo vyvinout systém, který měl uživateli umožnit sledovat a pracovat s naměřenými fyzikálními veličinami. Systém měl být vyvinut s ohledem na budoucí rozšiřování, neboť se předpokládalo, že po dosažení cílů stanovených touto prací nebude vývoj ukončen, ale bude aktivně pokračovat na základě získaných zkušeností z produkčního nasazení a dalších požadavků zákazníků. Zároveň měl být systém navržen tak, aby mohl postupně nahradit zastaralé monitorovací systémy nasazené u různých zákazníků, které byly vyvíjeny na míru konkrétnímu použití a jejich údržba a podpora byla velmi neefektivní.

V rámci práce vznikl systém, který plnohodnotně implementuje všechny požadované funkce, při vývoji byly zároveň zohledněny oba výše zmíněné požadavky. Vzhledem k prodloužení termínu odevzdání práce byl systém již nasazen do produkčního provozu a je tedy možné zmínit získané zkušenosti. I přes některé chyby, které doprovázely vydání první produkční verze, byl systém mezi uživateli přijat pozitivně. Potvrdilo se tvrzení v zmíněné v kapitole o analýze řešení, že někteří uživatelé zprovoznili systém pouze do takové míry, aby byly splněny interní předpisy. Jiní uživatelé naopak využívají naplno a zároveň poskytují zpětnou vazbu, případně požadují další rozšíření. Díky zpětné vazbě bylo také možné odladit několik chyb, které se i přes pečlivé testování dostaly do produkční verze.

Přínosem této práce je kromě obohacení portfolia firmy CUTTER Systems spol. s r.o. i rozšíření samotného jádra CIS a dalších komponent. Zároveň byl nastaven nový standard vývoje CIS modulů, neboť SCADA modul byl prvním modulem nové generace CIS postavené na WPF frameworku a také byl prvním projektem založeném na MVVM architektuře. V budoucnu čeká práci další rozšiřování, například migrace serverové části modulu (spolu se serverovou částí CIS) na *.Net Standard*. Serverová část se tak stane nezávislou na platformě a bude tak moci v rámci „krabicového“ řešení být instalována například na *Raspberry Pi*. Pro klientskou část je výhledově plánováno vytvoření webové aplikace, která alespoň v omezené míře poskytne přehled nejdůležitějších informací a může tak sloužit jako alternativa desktopové aplikace.

Osobně mi práce přinesla zkušenosti s vývojem systému od fáze návrhu přes vývoj a odladění až po následnou podporu po nasazení do produkčního prostředí.

Literatura

[1] Stuart A. Boyer. *SCADA: Supervisory Control and Data Acquisition, Fourth Edition*. 2010 [cit. 2019-02-06].

[2] David Bailey, Edwin Wright. *Practical SCADA for Industry*. 2003 [cit. 2019-02-06].

[3] Martin Fowler. *Patterns of Enterprise Application Architecture*. 2012 [cit. 2018-12-13].

[4] Matthew D. Groves. *AOP in .NET*. 2013 [cit. 2018-12-13].

[5] Ryan Vice, Muhammad Shujaat Siddiqi. *MVVM Survival Guide for Enterprise Architectures in Silverlight and WPF*. 2012 [cit. 2019-02-19].

[6] *Kahanu: Dynamic Expressions*, GitHub [online]. [cit. 2018-11-16]. Dostupné z: `https://github.com/kahanu/System.Linq.Dynamic/wiki/Dynamic-Expressions`

[7] *C# Guide*, Microsoft [online]. 2018 [cit. 2018-11-15]. Dostupné z: `https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/`

[8] *Xceed Toolkit Plus for WPF*, Xceed Software Inc [online]. 2018 [cit. 2019-02-22]. Dostupné z: `https://xceed.com/wp-content/documentation/xceed-toolkit-plus-for-wpf/webframe.html#Welcome.html`

[9] *PostgreSQL*, The PostgreSQL Global Development Group [online]. 2018 [cit. 2018-12-18]. Dostupné z: `https://www.postgresql.org/`

[10] *OPC*, OPC Foundation [online]. 2018 [cit. 2018-12-28]. Dostupné z: `https://opcfoundation.org/`

A Obsah CD s přílohami

Přiložené CD obsahuje následující adresáře:

1. *database* - záloha předpřipravené databáze systému
2. *dist* - soubory potřebné pro zprovoznění instalace
3. *doc* - elektronická verze tohoto dokumentu.
4. *src* - zdrojové soubory práce
   (a) client - klientská část CIS frameworku
   (b) dependencies - podpůrné knihovny
   (c) modules - SCADA modul
   (d) server - serverová část CIS frameworku